ECS171: Machine Learning

L14: Unsupervised Learning I

Instructor: Prof. Maike Sonnewald
TAs: Pu Sun & Devashree Kataria

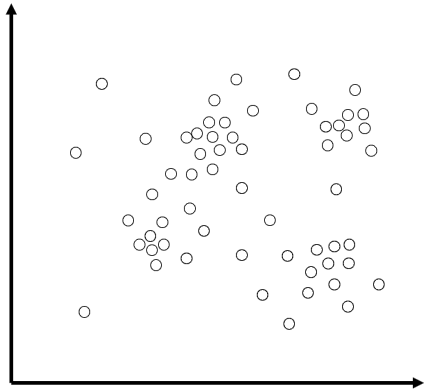

MOCO Amsterdam garden

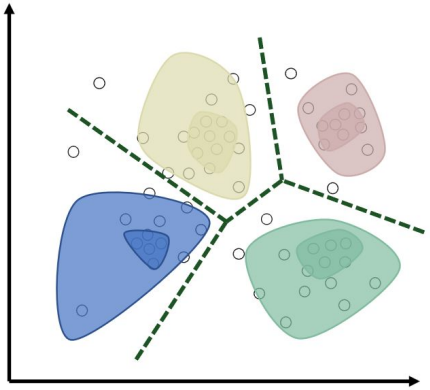
Intended Learning Outcomes

- Appreciate and describe the difference between supervised and unsupervised learning
- Describe different unsupervised types (Latent, Association, Clustering)
- Describe generally different types of clustering methods and how they relate to each other and the data-distributions
- Describe and apply k-means clustering and Fuzzy c-Means clustering
- Describe the importance and features of various distance metrics and where each is appropriate and what assumptions are made (e.g. linearity)
- Describe hierarchical and density based clustering

Unsupervised

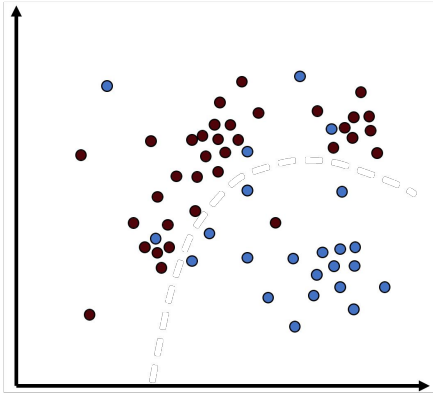
- No labels
- Identify structures
- **No explicit feedback**

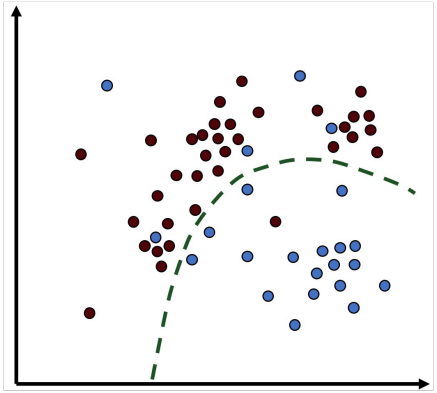




Supervised

- Labeled data
- Decision boundary





Training data

Resulting model

[Langs+ 2018]

Unsupervised learning

Tasks to consider:

- Reduce dimensionality
- Find clusters
- Model data density
- Find hidden causes

Key utility:

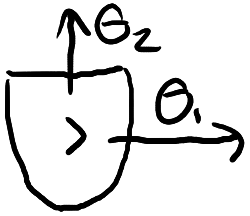
- Compress data
- Detect outliers
- Facilitate other learning

R. Grosse

Unsupervised approaches

- Latent variable models:
  - Assume data depends on some latent variables that are never observed. Such models are called latent variable models
  - Can be used to approximate high dimensional data using lower dimensional form
  - Examples: PCA and Autoencoders
- Association
  - Checks for the dependency of one data item on another data item and maps accordingly
  - Tries to find some interesting relations or associations among the variables of dataset
  - Based on different rules to discover relations between variables in the database
  - Used extensively in retain, e.g., so that it can be more ‘profitable’
- Clustering (focus today)
  - Partitioning of the data
  - It's well-suited for processes such as customer segmentation, exploratory data analysis or image recognition

latents: cluster membership



θ_1 = -90° -45 0 45 90

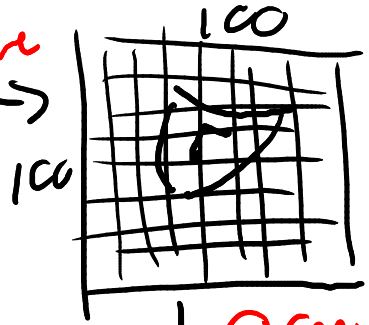
θ_2 -90° -45 0 45 90

① Orient mask

$\theta_1 = -45$

$\theta_2 = -45$

② take a picture

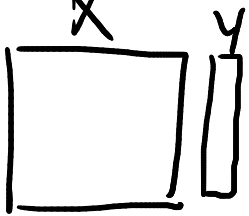


③ convert to vector

θ_1, θ_2 10,000

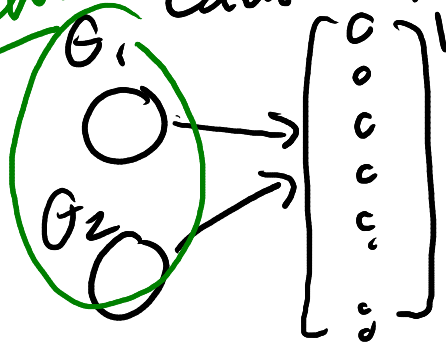
"raw" "data" vs. "intrinsic" "latent"

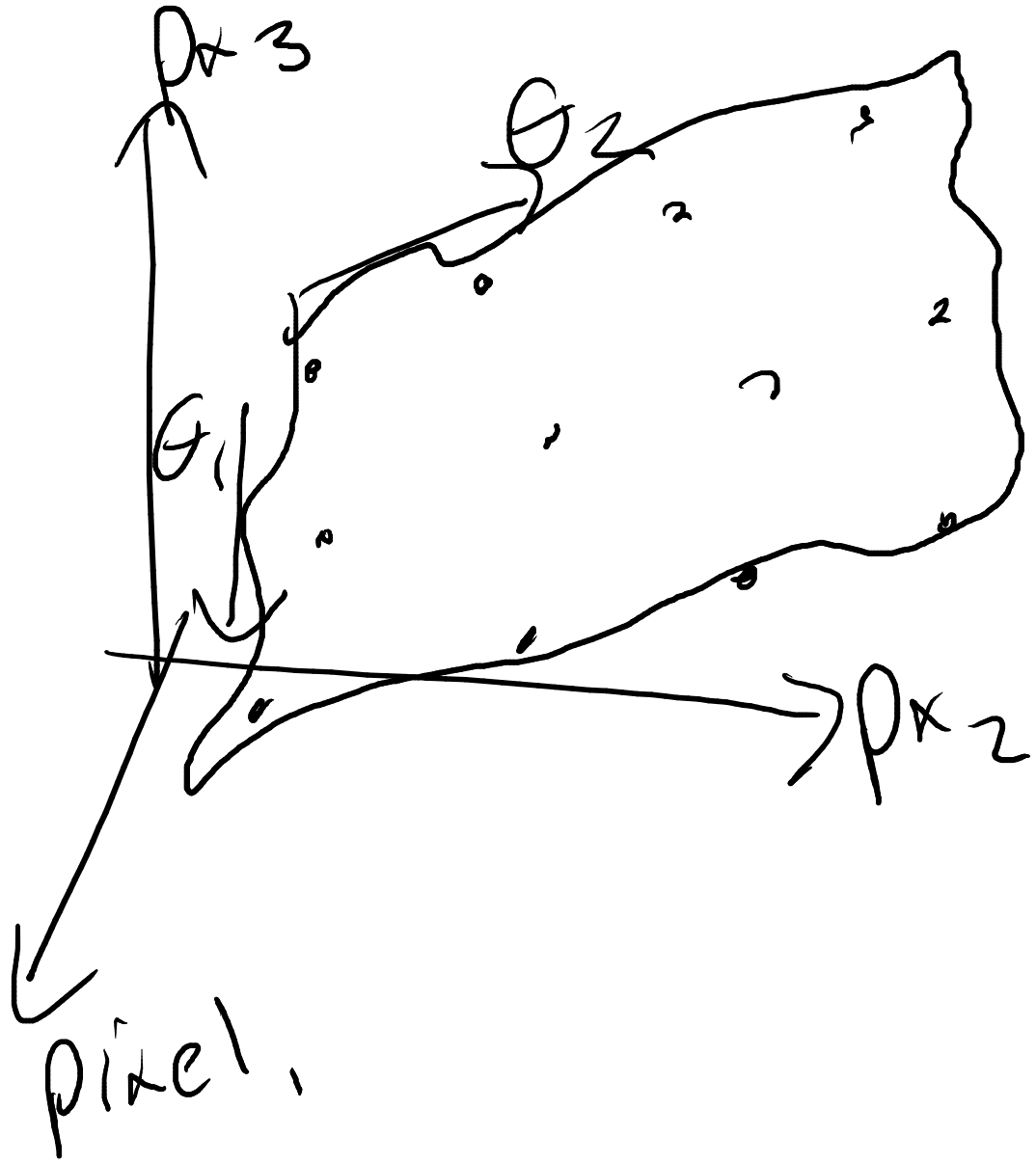
What's the dimensionality?



latent

Causal Graph




px3
θ2
θ1
px2
pixel.

Clustering

Unsupervised learning is a method used to identify natural groupings within unlabeled and unstructured data

It clusters objects based on their similarity, grouping together objects that are more similar to each other than to objects in other clusters.

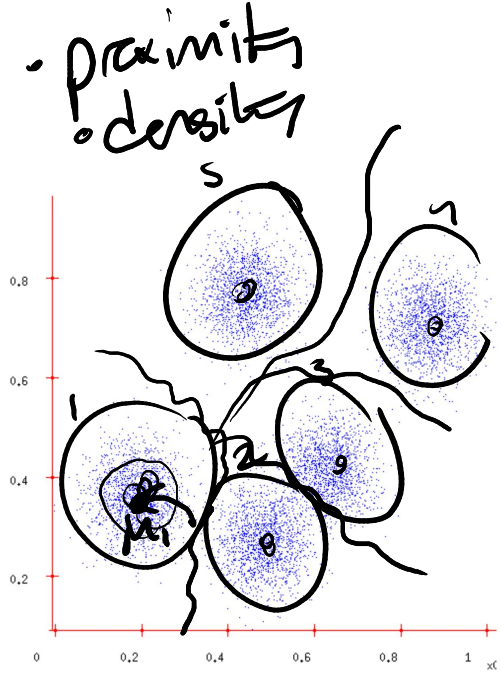
Parameters to create maximally homogeneous groupings:

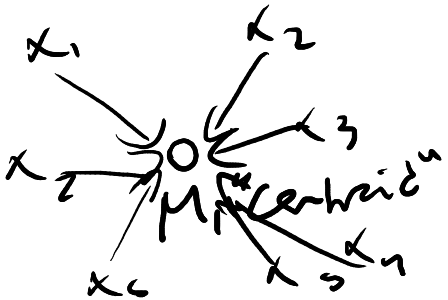
- ~~A proper subset of features~~
- Parameter and unsupervised methodology choice

Goal: To model the underlying structure in the data in order to learn more about the data

Clustering

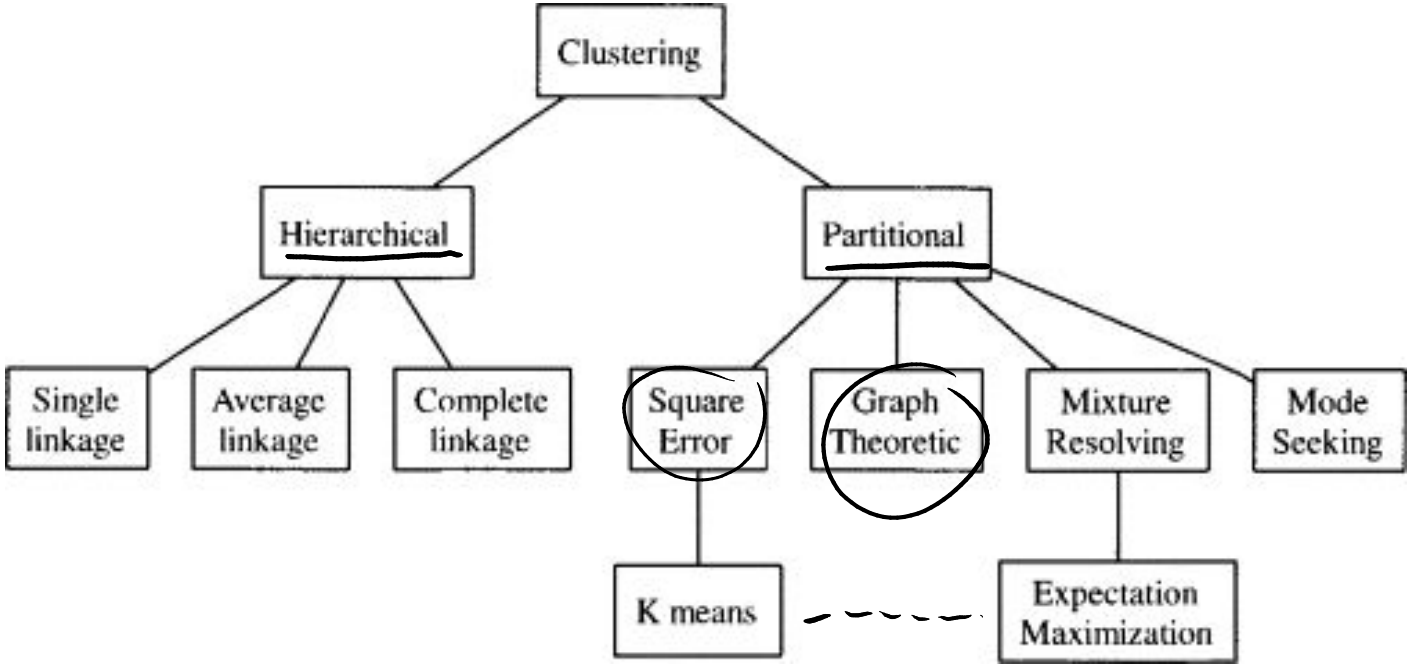
- Grouping N examples into K clusters one of canonical problems in unsupervised learning
- Motivation: prediction; lossy compression; outlier detection
- We assume that the data was generated from a number of different classes
- The aim is to cluster data from the same class together.
  - How many clusters?
  - Why not put each datapoint into a separate cluster?
- What is the objective function that is optimized by sensible clustering?





$$\sum_{i=1}^{n} dist(\mu_1, x_i)$$

Types of clustering methods (non-exhaustive)

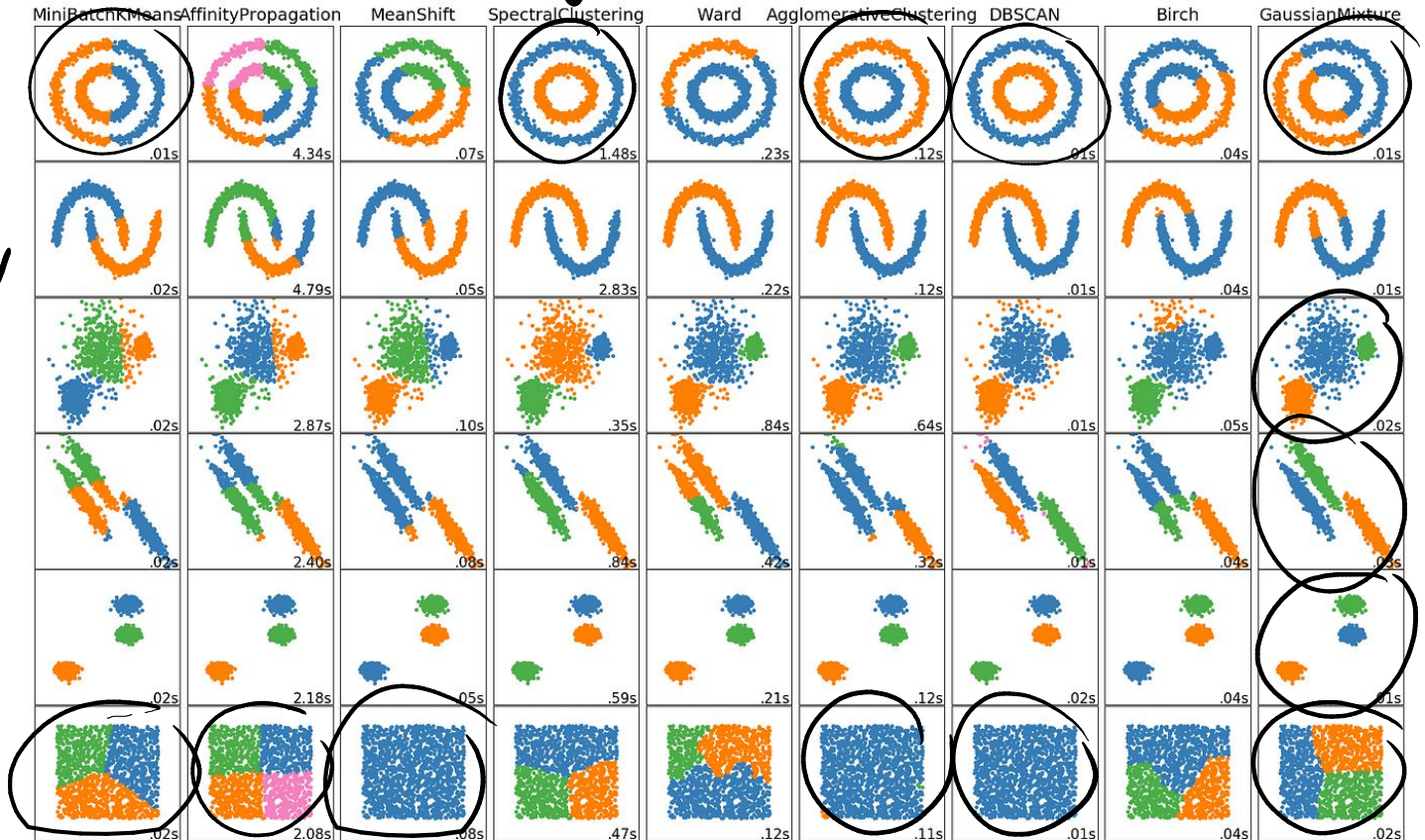


What is the objective function that is optimized by sensible clustering?

scikit-learn

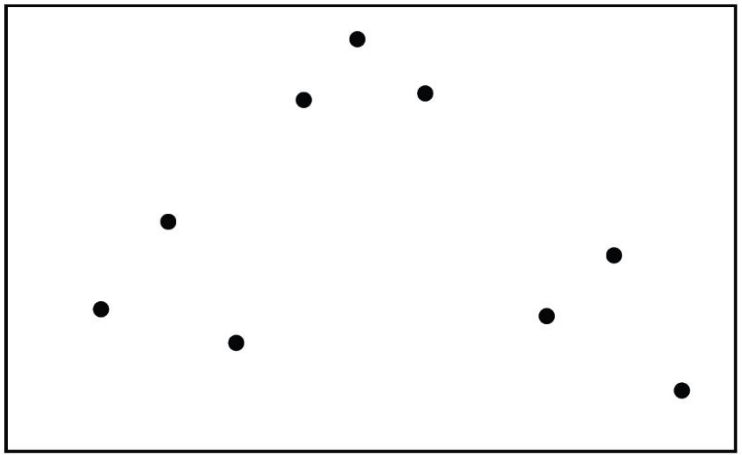
algo →

datasets



Clustering

- Assume the data {x(1), . . . , x(N)} lives in a Euclidean space, x(n) ∈ R^d
- Assume the data belongs to K classes (patterns) clusters
- How can we identify those classes (data points that belong to each class)?

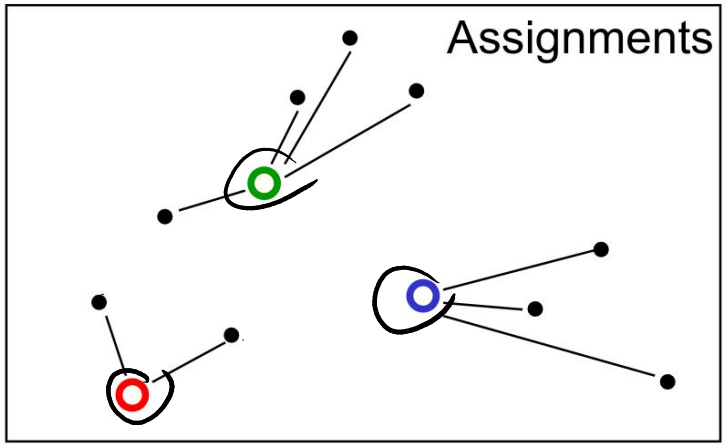


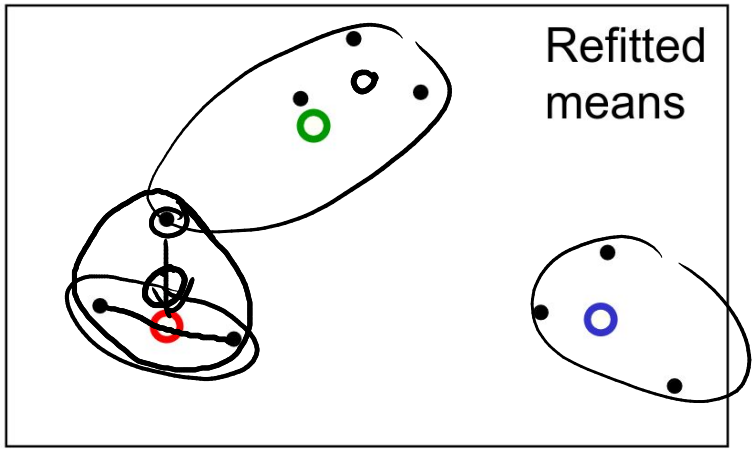
K-means

Clustering

choose K K=3

- Initialization: randomly initialize cluster centers
- The algorithm iteratively alternates between two steps:
  - Assignment step: Assign each data point to the closest cluster
  - Refitting step: Move each cluster center to the ~~center of gravity~~ Euclidean center of the data assigned to it





Voronoi Partition

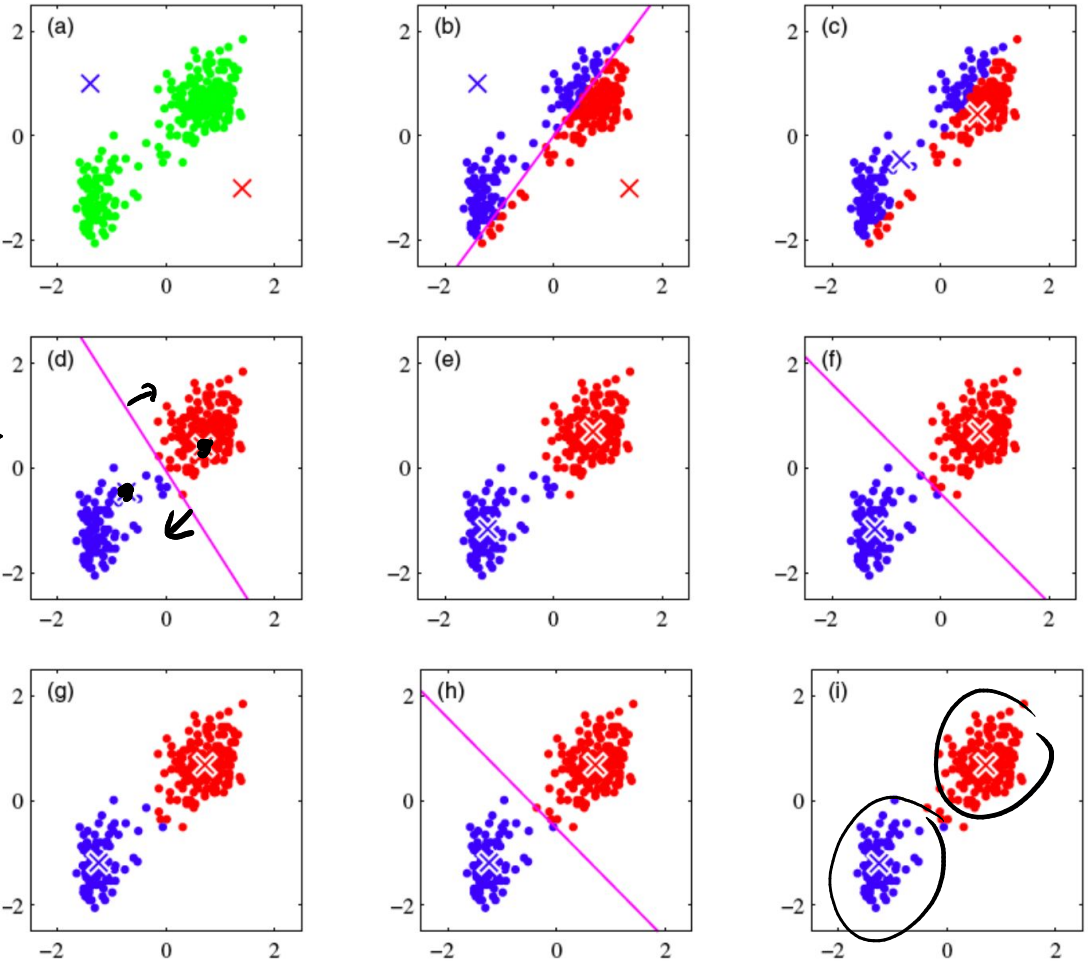


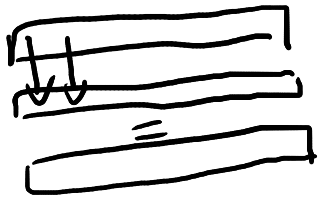
Figure from Bishop

Simple demo: http://syskall.com/kmeans.js/

Cluster Centroid (Middle of a cluster)

- Cluster centroid μ is considered as a measure of cluster location
  - It represents the center point of a cluster
  - Centroid is the multi-dimensional average of the cluster
- Each centroid represents the "average observation" within a cluster across all the attributes in the analysis

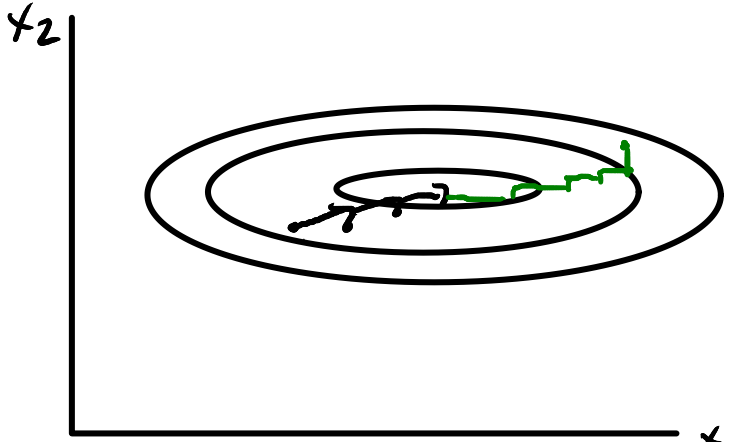
$$\mu_j = \frac{1}{|c_j|} \sum_{x^{(i)} \in c_j} x^{(i)}$$



- Example: $X = \{(80, 56), (75, 53), (60, 50), (68,54)\}$, $X \in c_j$,

$$\mu_j = \left(\frac{80+75+60+68}{4}, \frac{56+53+50+54}{4}\right) = (71.75,\ 53.25)$$

k-Means



Uses ‘euclidean’ distance metric

Is ‘exclusive’ clustering

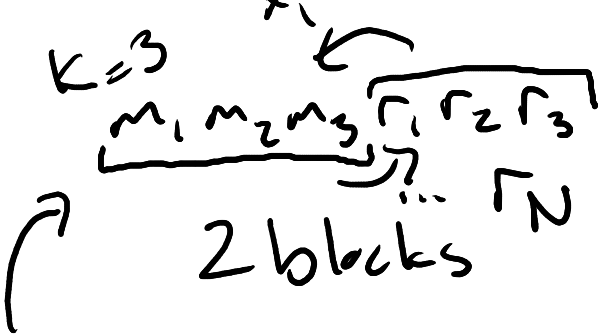
What is actually being optimized?

K-means Objective:
Find cluster centers $\{\mathbf{m}\}$ and assignments $\{\mathbf{r}\}$ to minimize the sum of squared distances of data points $\{\mathbf{x}^{(n)}\}$ to their assigned cluster centers

$$\{\mathbf{m}\}, \{\mathbf{r}\} = \arg\min_{\{\mathbf{m}\},\{\mathbf{r}\}} J(\{\mathbf{m}\}, \{\mathbf{r}\}) = \arg\min_{\{\mathbf{m}\},\{\mathbf{r}\}} \sum_{n=1}^{N} \sum_{k=1}^{K} r_k^{(n)} \|\mathbf{m}_k - \mathbf{x}^{(n)}\|^2$$

$$\text{s.t.} \sum_k r_k^{(n)} = 1, \forall n, \quad \text{where} \quad r_k^{(n)} \in \{0,1\}, \forall k, n$$

where $r_k^{(n)} = 1$ means that $\mathbf{x}^{(n)}$ is assigned to cluster k (with center $\mathbf{m}_k$)



Optimization method is a form of coordinate descent (‘block coordinate descent’)

- Fix centers, optimize assignments (choose cluster whose mean is closest)
- Fix assignments, optimize means (average of assigned data points)

K-means

cluster guess data Distance function

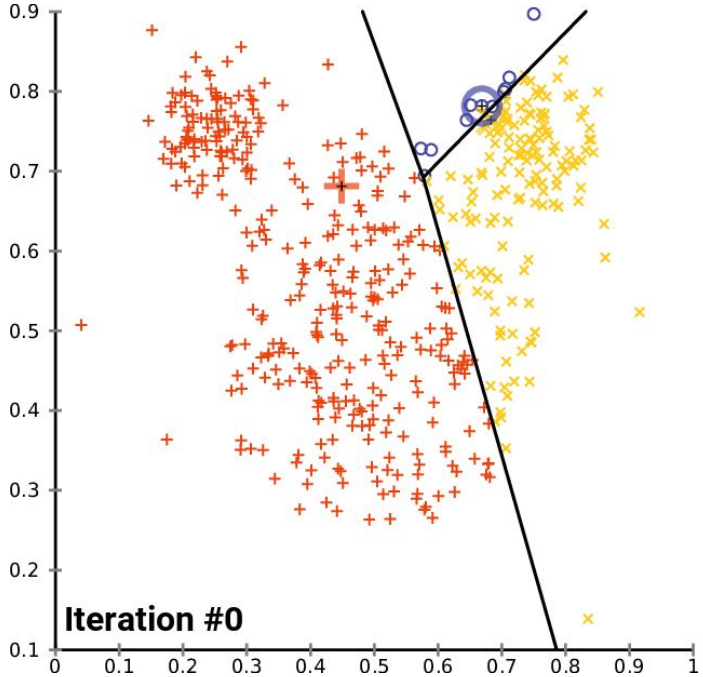
$$J = \sum_{j=1}^{K} \sum_{x^{(i)} \in c_j} (x^{(i)} - \mu_j)^2$$

data point centroids

Minimize objective function

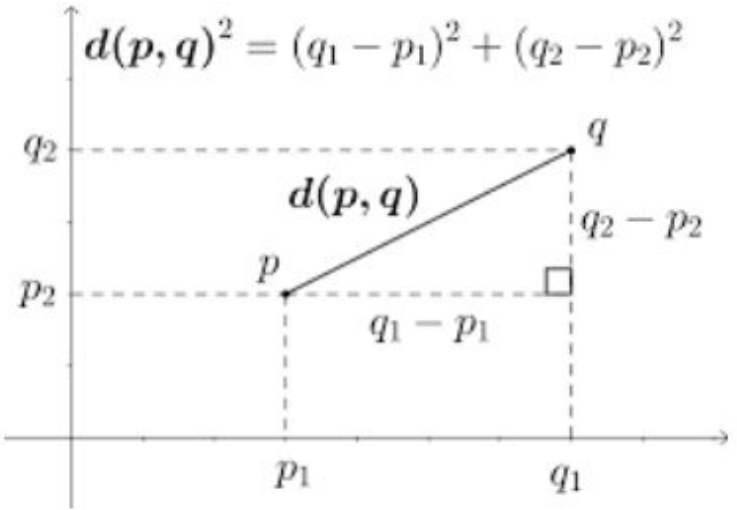
Setting K the number of clusters

Stochastic first guess



Euclidean ‘Distance’

- Euclidean Distance is the length of a line segment between two points in the Euclidean space.



$$d(\mathbf{p},\mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}.$$

k-Means: Iteration 1

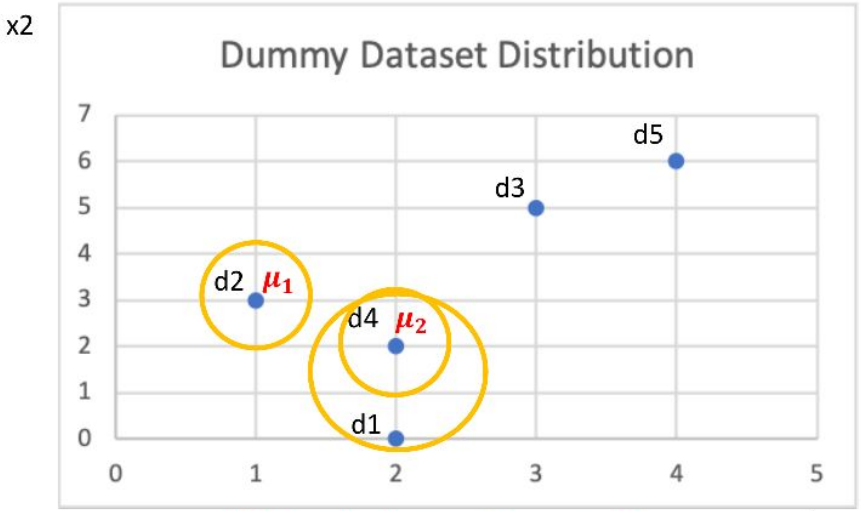
- Assume k=2
- Choose two data points (d2 , d4) randomly and use as the centroids of two clusters
- Calculate the Euclidean distance of the next data point (say d1) from each of the two cluster centroids
- The new data point joins the cluster with the minimum distance.

$$d_{Euclidean}(\mu_1, d1) = \sqrt{(1-2)^2 + (3-0)^2}$$
$$d_{Euclidean}(\mu_1, d1) = 3.16$$

$$d_{Euclidean}(\mu_2, d1) = \sqrt{(2-2)^2 + (2-0)^2}$$
$$d_{Euclidean}(\mu_2, d1) = 2$$

$$min\left(d_{Euclidean}(\mu_1, d1), d_{Euclidean}(\mu_2, d1)\right) = 2$$

	x1	x2
d1	2	0
d2	1	3
d3	3	5
d4	2	2
d5	4	6



d1 joins the cluster with centroid μ_2.

k-Means: Iteration 1

	x1	x2
d1	2	0
d2	1	3
d3	3	5
d4	2	2
d5	4	6

- Since the shape of the cluster with centroidm μ_2 has changed, a new centroid is calculated

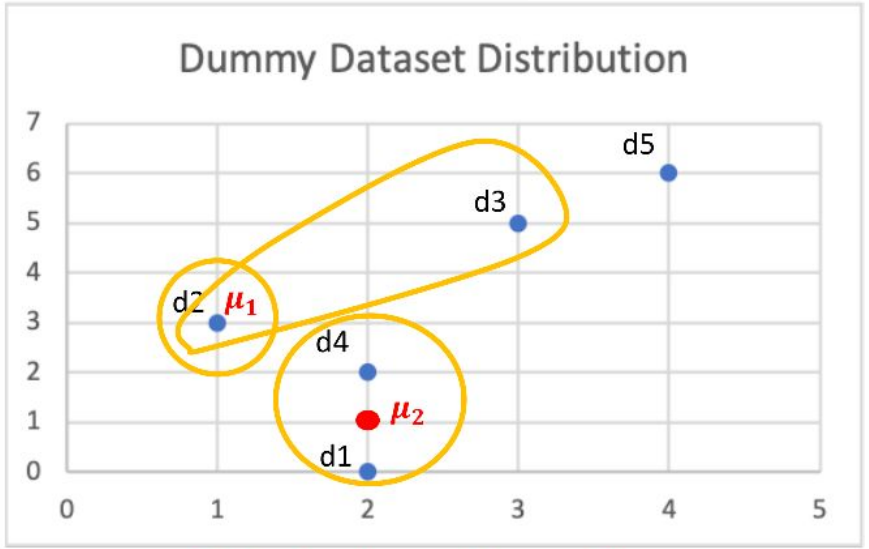
$$\mu_2 = (\frac{2+2}{2}, \frac{0+2}{2}) = (2,1)$$

- Repeat the same process for the nex point (i.e. d3)

$$d_{Euclidean}(\mu_1, d3) = \sqrt{(1-3)^2 + (3-5)^2}$$
$$d_{Euclidean}(\mu_1, d3) = 2.83$$

$$d_{Euclidean}(\mu_2, d3) = \sqrt{(2-3)^2 + (1-5)^2}$$
$$d_{Euclidean}(\mu_2, d3) = 4.12$$

$$min\left(d_{Euclidean}(\mu_1, d3), d_{Euclidean}(\mu_2, d3)\right) = 2.83$$



d3 joins the cluster with centroid μ_1.

k-Means: Iteration 1

	x1	x2
d1	2	0
d2	1	3
d3	3	5
d4	2	2
d5	4	6

- Since the shape of the cluster with centroid μ_1 has changed, a new centroid has to be calculated for this cluster.

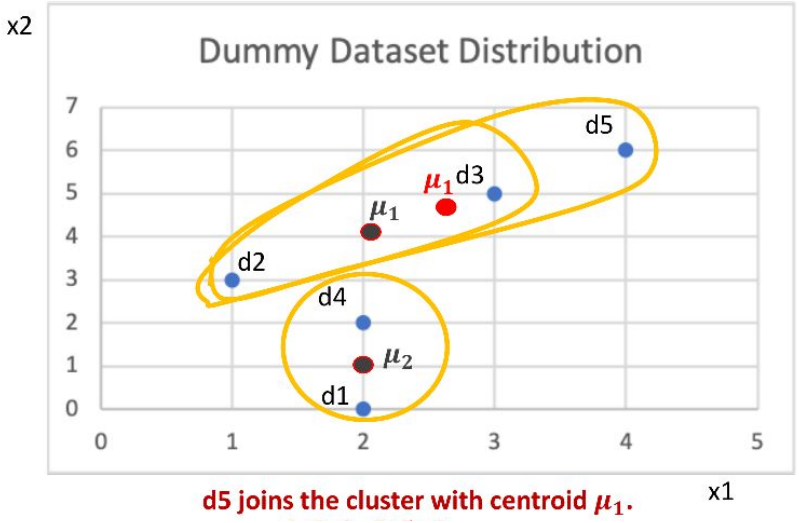
Updating μ_1 : $\mu_1 = (\frac{1+3}{2}, \frac{3+5}{2}) = (2,4)$

- Repeat the same process for the next data point (i.e. d5)

$$d_{Euclidean}(\mu_1, d5) = \sqrt{(2-4)^2 + (4-6)^2}$$
$$d_{Euclidean}(\mu_1, d5) = 2.82$$

$$d_{Euclidean}(\mu_2, d5) = \sqrt{(2-4)^2 + (1-6)^2}$$
$$d_{Euclidean}(\mu_2, d5) = 5.38$$

$$min\left(d_{Euclidean}(\mu_1, d5), d_{Euclidean}(\mu_2, d5)\right) = 2.82$$



d5 joins the cluster with centroid μ_1.

k-Means: Iteration 1

	x1	x2
d1	2	0
d2	1	3
d3	3	5
d4	2	2
d5	4	6

- Since the shape of the cluster with centroid μ_1 has changed, a new centroid has to be calculated for this cluster.

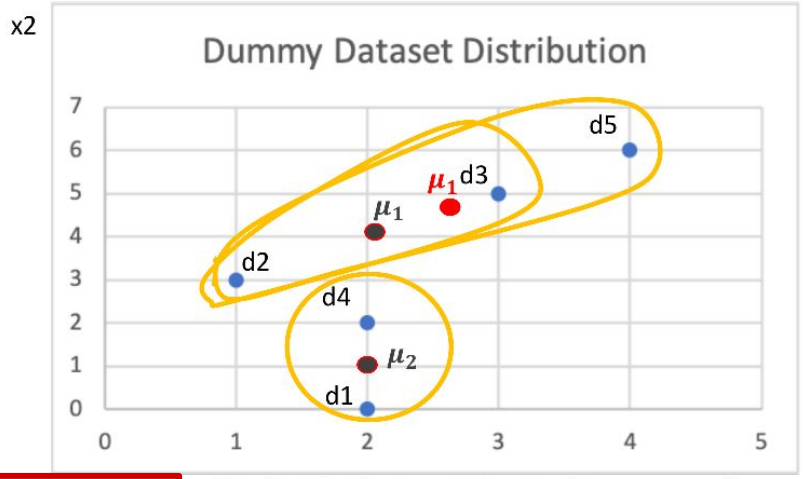
Updating μ_1 : $\mu_1 = (\frac{1+3}{2}, \frac{3+5}{2})$ = (2,4)

- Repeat the same process for the next data point (i.e. d5)

$$d_{Euclidean}(\mu_1, d5) = \sqrt{(2-4)^2 + (4-6)^2}$$
$$d_{Euclidean}(\mu_1, d5) = 2.82$$

$$d_{Euclidean}(\mu_2, d5) = \sqrt{(2-4)^2 + (1-6)^2}$$
$$d_{Euclidean}(\mu_2, d5) = 5.38$$

$$min\left(d_{Euclidean}(\mu_1, d5), d_{Euclidean}(\mu_2, d5)\right) = 2.82$$



d5 joins the cluster with centroid μ_1.

Iteration 1 complete

Updating μ_1 : $\mu_1 = (\frac{1+3+4}{3}, \frac{3+5+6}{3})$ = (2.67,4.67)

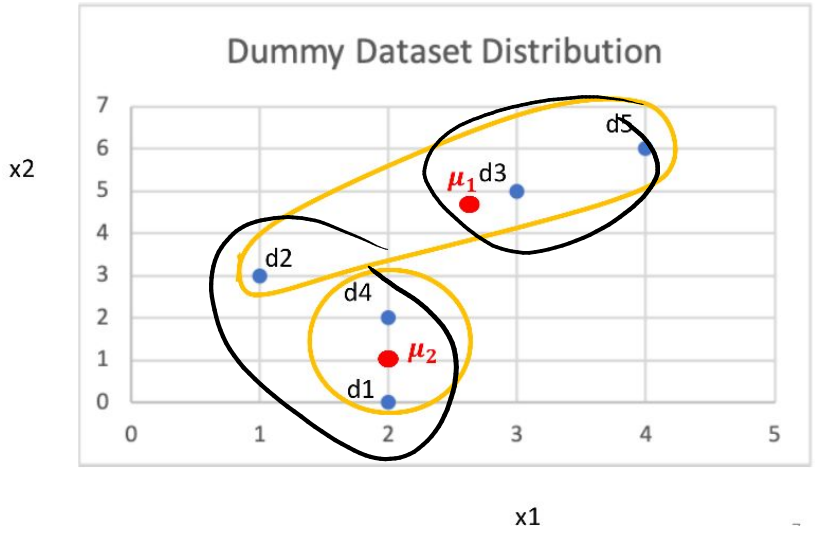
k-Means: Iteration 2

- In iteration 2, the distances are calculated from the new centroids (i.e., μ_1 and μ_2)
- μ_1 = (2.67,4.67)
- μ_2 = (2, 1)

$$d_{Euclidean}(\mu_j, di)$$

	Euclidean distance from μ_1	Euclidean distance from μ_2
d1	4.7	1
d2	2.37	2.24
d3	0.46	4.13
d4	2.76	1
d5	1.89	5.39

	x1	x2
d1	2	0
d2	1	3
d3	3	5
d4	2	2
d5	4	6



k-Means: Iteration 2

- In iteration 2, the distances are calculated from the new centroids (i.e., μ_1 and μ_2)

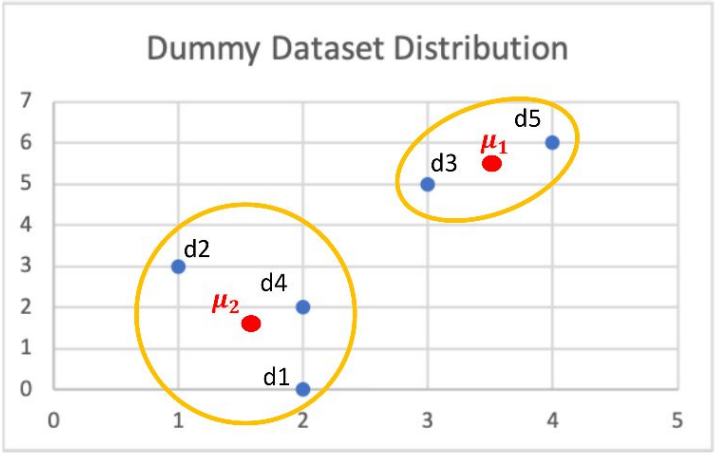
$\mu_1 = (\frac{3+4}{2}, \frac{5+6}{2}) = (3.5, 5.5)$

$\mu_2 = (\frac{2+1+2}{3}, \frac{0+3+2}{3}) = (1.7, 1.7)$

$d_{Euclidean}(\mu_j, di)$

	x1	x2
d1	2	0
d2	1	3
d3	3	5
d4	2	2
d5	4	6

	Euclidean distance from μ_1	Euclidean distance from μ_2
d1	4.7	1
d2	2.37	2.24
d3	0.46	4.13
d4	2.76	1
d5	1.89	5.39



k-Means: Iteration 3 [clusters are now exclusive]

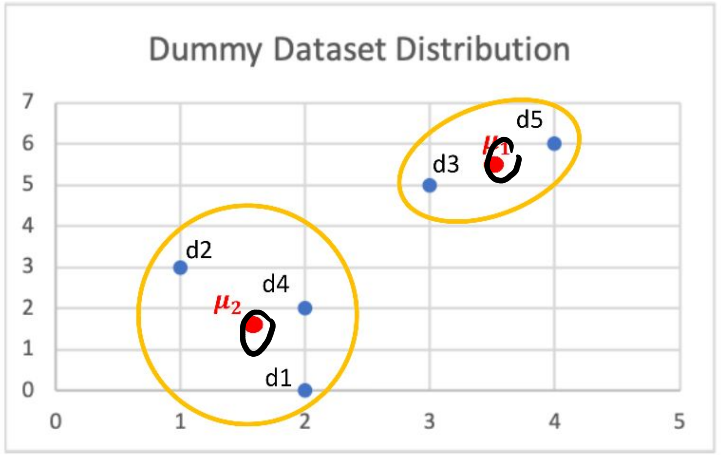
- In iteration 3, the distances are calculated from the new centroids (i.e., μ_1 and μ_2)
- μ_1=(3.5,5.5)
- μ_2= (1.7, 1.7)

	x1	x2
d1	2	0
d2	1	3
d3	3	5
d4	2	2
d5	4	6

$$d_{Euclidean}(\mu_j, di)$$

	Euclidean distance from μ_1	Euclidean distance from μ_2
d1	5.7	1.7
d2	3.53	1.47
d3	0.7	3.5
d4	3.8	0.42
d5	0.7	4.9

2



K-means for Vector Quantization

- Given image, construct “dataset“ of pixels represented by their RGB pixel intensities
- Run k-means, replace each pixel by its cluster center

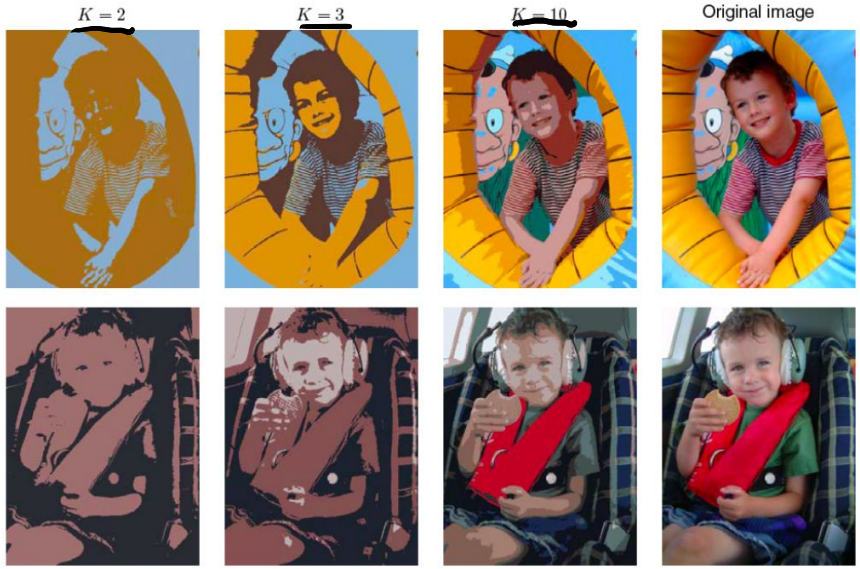
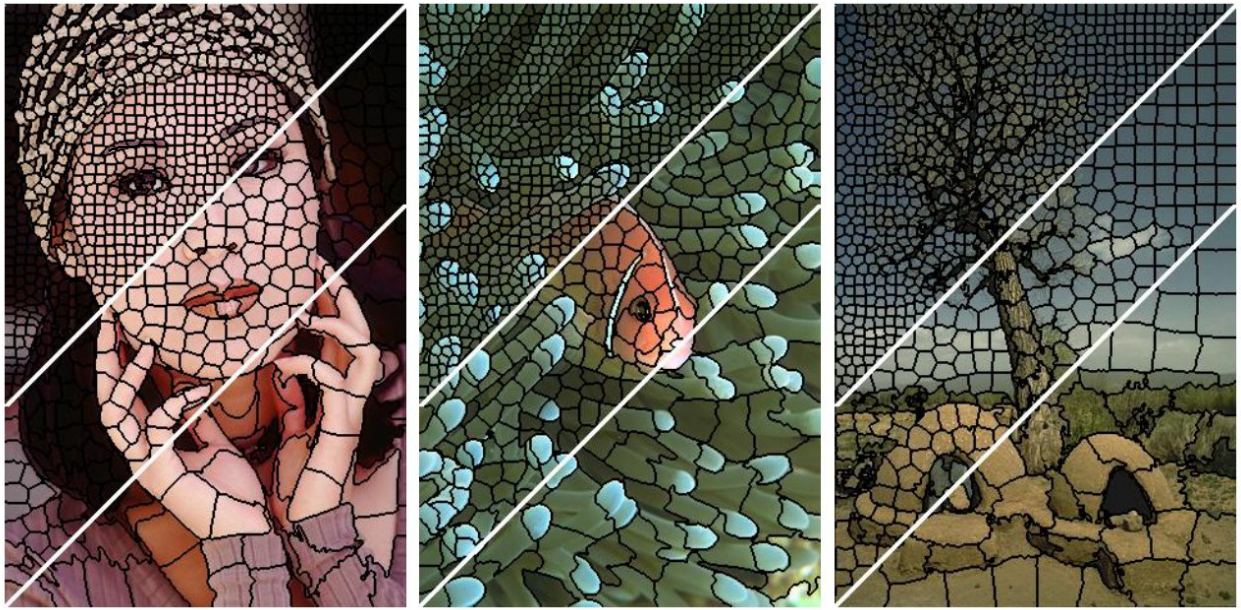


Figure from Bishop

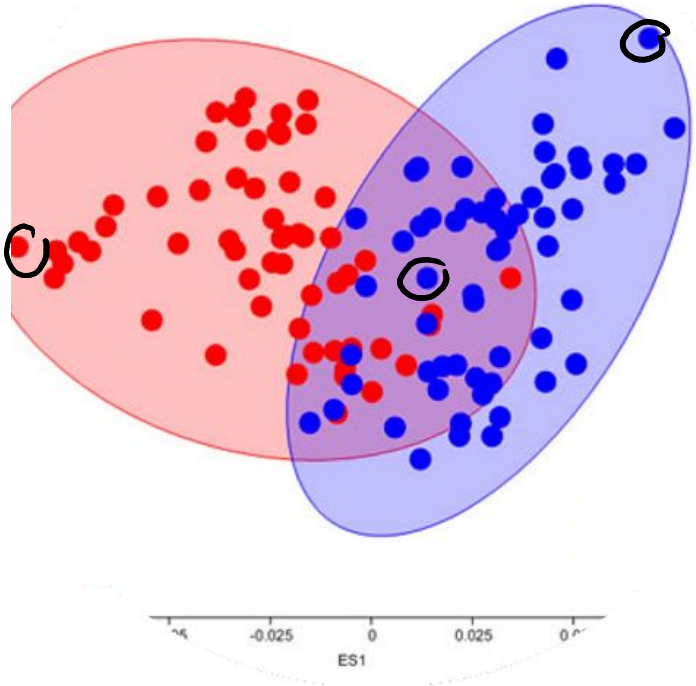
K-means for Image Segmentation

- Given image, construct “dataset“ of pixels represented by their RGB pixel intensities
- Run k-means (with some modifications) to get superpixels

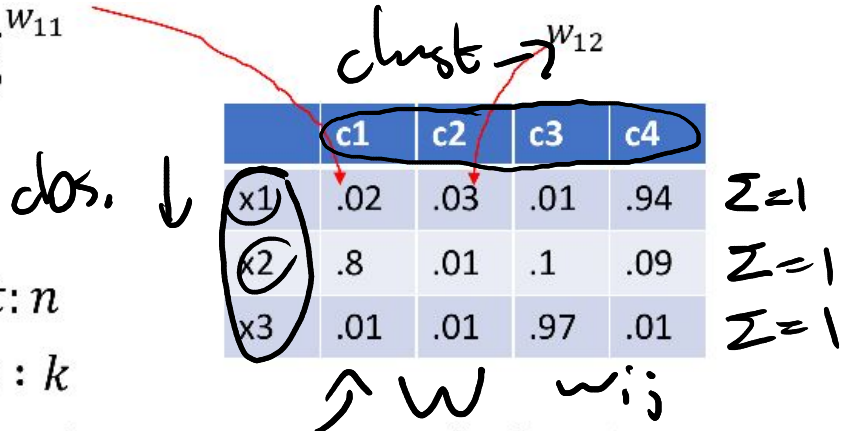


Fuzzy C-means (FCM)

- Not exclusive clustering
  - Data-points can belong to several clusters
- It carries the notion of fuzzy partitions where the **probability** of a data point *i* belonging to a cluster *j* (i.e. probability of membership) is a w_{ij} s.t $0 \leq w_{ij} \leq 1$.



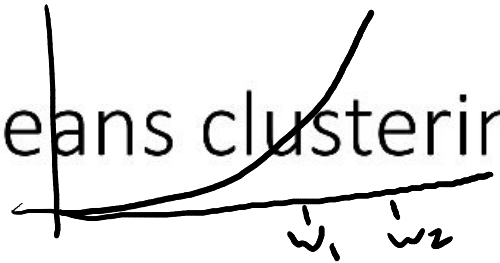
Fuzzy C-means Clustering

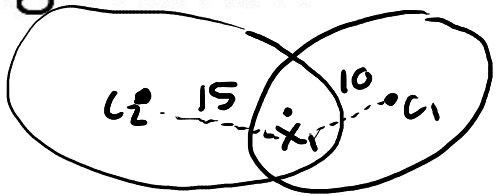


Assume:

- Data $X = \{x_1, \dots, x_n\}$; $size\ of\ the\ dataset: n$
- Clusters = $\{c_1, \dots, c_k\}$; $number\ of\ clusters: k$
- Probability of membership of data point x_i to cluster c_j is w_{ij} $such\ that$ $0 \leq w_{ij} \leq 1$.
- Example: suppose $X = \{x_1, x_2, x_3\}$, C = k =4
- Condition:
  1. All the weights for a given data point x_i should add up to 1: $\sum_{j=1}^{k} w_{ij} = 1$
  2. Each cluster c_j with non-zero weight, contains at least one data , but does not contain all data with a weight of 1: $0 < \sum_{i=1}^{n} w_{ij} < n$.

Fuzzy C-Means clustering (FCM) Algorithm





Input : Data, k , Output: w_{ij} , c_j, $1 \le j \le k$, $1 \le i \le n$

1. Initialization: Randomly select values for all w_{ij} s.t. $\sum_{j=1}^{k} w_{ij} = 1$

$w_{11} = 0.6$

$w_{12} = 0.4$

2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$; if p=0 , it is k-means , or else if p> 1, then we do more fuzzification. Lets say p = 2. **p serves as the fuzzifier.**

softmax

3. Update the fuzzy-pseudo partition, i.e., w_{ij} : $w_{ij} = \frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$ if p = 2,

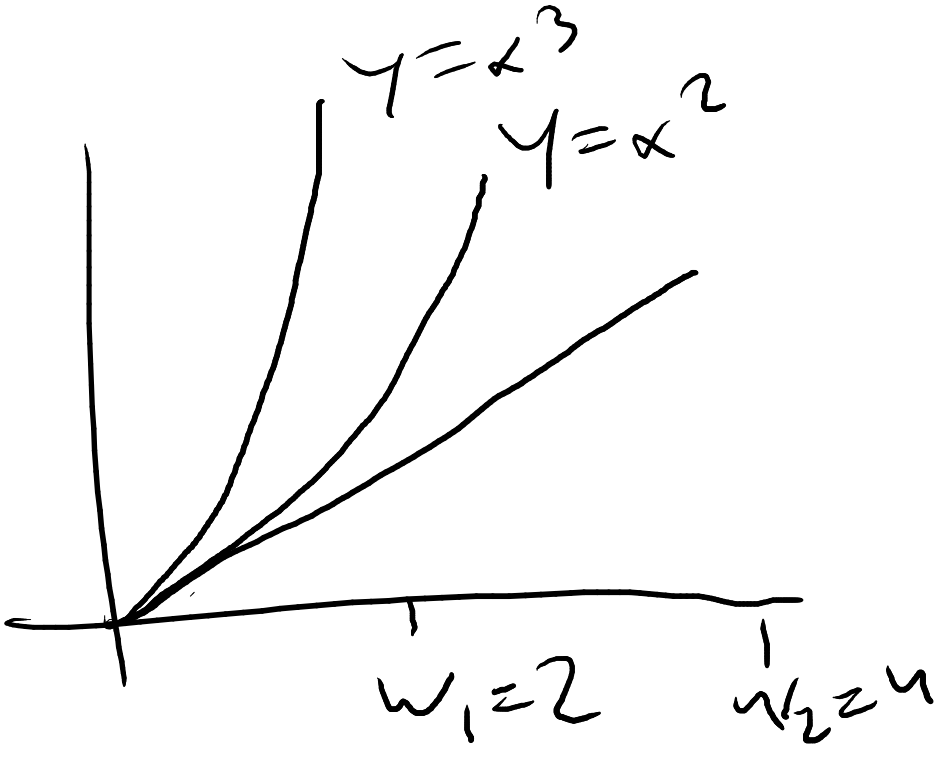
then $w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$

$\frac{1/10}{1/10 + 1/15}$ $\frac{1/15}{1/10 + 1/15}$

$SSE = \sum_{j=1}^{k}\sum_{i=1}^{n} w_{ij}^{p} dist(x_i, c_j)$

P > 1

4. Repeat step ~~2 and 3 until~~ w_{ij} doesn't change. (very much)



$$\frac{w_2}{w_1} = 2$$

$$\frac{w_2^2}{w_1^2} = \frac{16}{4} = 4$$

$$\frac{w_2^3}{w_1^3} = \frac{64}{8} = 8$$

$w_1 = 0.3$
$w_2 = 0.7$

$$\frac{0.7}{0.3}$$

$$\frac{0.7^2}{0.3^2}$$

$$\frac{0.7^3}{0.3^3}$$

Fuzzy C-Means clustering (FCM) : Example

Input : Data :{(1,2) , (2,3) , (9.4), (10,1)}. , k=2 ,

Output: w_{ij} , $c_j, 1 \leq j \leq k$, $1 \leq i \leq n$

1. Initialization: Randomly select values for all w_{ij} *s.t.* $\sum_{j=1}^{k} w_{ij} = 1$
2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$;p=2.
3. Update the fuzzy-pseudo partition, w_{ij} : w_{ij} = $\frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$, if p = 2, then w_{ij} = $\frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$
4. Repeat step 2 and 3 until w_{ij} doesn't change (convergence).

Step 1: Randomly initialize weights

	c1	c2
x1	.4	.6
x2	.88	.12
x3	.41	.59
x4	.27	.73

Data

	att1	att2
x1	1	2
x2	2	3
x3	9	4
x4	10	1

Fuzzy C-Means clustering (FCM) : Example

Input : Data :{(1,2) , (2,3) , (9.4), (10,1)}. , k=2 ,

Output: w_{ij} , c_j, $1 \le j \le k$, $1 \le i \le n$

1. Initialization: Randomly select values for all w_{ij} *s.t.* $\sum_{j=1}^{k} w_{ij} = 1$
2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$;p=2.
3. Update the fuzzy-pseudo partition, w_{ij} : $w_{ij} = \frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$, if p = 2, then $w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$
4. Repeat step 2 and 3 until w_{ij} doesn't change (convergence).

Data

	att1	att2
x1	1	2
x2	2	3
x3	9	4
x4	10	1

Step 1: Randomly initialize weights

	c1	c2
x1	.4	.6
x2	.88	.12
x3	.41	.59
x4	.27	.73

Step 2:

j=1 ➔ $c_1 = \frac{\sum_{i=1}^{n} w_{i1}^{2} x_i}{\sum_{i=1}^{n} w_{i1}^{2}}$, j=2 ➔ $c_2 = \frac{\sum_{i=1}^{n} w_{i2}^{2} x_i}{\sum_{i=1}^{n} w_{i2}^{2}}$

Fuzzy C-Means clustering (FCM) : Example

Input : Data :{(1,2) , (2,3) , (9.4), (10,1)}. , k=2 ,

Output: w_{ij} , $c_j, 1 \leq j \leq k$, $1 \leq i \leq n$

1. Initialization: Randomly select values for all w_{ij} *s.t.* $\sum_{j=1}^{k} w_{ij} = 1$
2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$;p=2.
3. Update the fuzzy-pseudo partition, w_{ij} : w_{ij} = $\frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$, if p = 2, then $w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$
4. Repeat step 2 and 3 until w_{ij} doesn't change (convergence).

Data

	att1	att2
x1	1	2
x2	2	3
x3	9	4
x4	10	1

Step 1: Randomly initialize weights

	c1	c2
x1	.4	.6
x2	.88	.12
x3	.41	.59
x4	.27	.73

Step 2:

$$j=1 \rightarrow c_1 = \frac{\sum_{i=1}^{n} w_{i1}^{2} x_i}{\sum_{i=1}^{n} w_{i1}^{2}}, \quad j=2 \rightarrow c_2 = \frac{\sum_{i=1}^{n} w_{i2}^{2} x_i}{\sum_{i=1}^{n} w_{i2}^{2}}$$

$(.4)^2+(.88)^2+(.41)^2+(.27)^2 = 1.18$

$(.6)^2+(.12)^2+(.59)^2+(.73)^2 = 1.25$

Fuzzy C-Means clustering (FCM) : Example

Input : Data :{(1,2) , (2,3) , (9.4), (10,1)}. , k=2 ,

Output: w_{ij} , $c_j, 1 \leq j \leq k$, $1 \leq i \leq n$

1. Initialization: Randomly select values for all w_{ij} *s.t.* $\sum_{j=1}^{k} w_{ij} = 1$
2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$;p=2.
3. Update the fuzzy-pseudo partition, w_{ij} : w_{ij} = $\frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$, if p = 2, then $w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$
4. Repeat step 2 and 3 until w_{ij} doesn't change (convergence).

Data

	att1	att2
x1	1	2
x2	2	3
x3	9	4
x4	10	1

Step 1: Randomly initialize weights

	c1	c2
x1	.4	.6
x2	.88	.12
x3	.41	.59
x4	.27	.73

Step 2:

$$j=1 \rightarrow c_1 = \frac{\sum_{i=1}^{n} w_{i1}^{2} x_i}{\sum_{i=1}^{n} w_{i1}^{2}}, \quad j=2 \rightarrow c_2 = \frac{\sum_{i=1}^{n} w_{i2}^{2} x_i}{\sum_{i=1}^{n} w_{i2}^{2}}$$

$(.4)^2+(.88)^2+(.41)^2+(.27)^2 = 1.18$

$(.6)^2+(.12)^2+(.59)^2+(.73)^2 = 1.25$

Fuzzy C-Means clustering (FCM) : Example

Input : Data :{(1,2) , (2,3) , (9.4), (10,1)}. , k=2 ,

Output: w_{ij} , $c_j, 1 \le j \le k$, $1 \le i \le n$

1. Initialization: Randomly select values for all w_{ij} *s.t.* $\sum_{j=1}^{k} w_{ij} = 1$
2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$;p=2.
3. Update the fuzzy-pseudo partition, w_{ij} : $w_{ij} = \frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$, if p = 2, then $w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$
4. Repeat step 2 and 3 until w_{ij} doesn't change (convergence).

Data

	att1	att2
x1	1	2
x2	2	3
x3	9	4
x4	10	1

Step 1: Randomly initialize weights

	c1	c2
x1	.4	.6
x2	.88	.12
x3	.41	.59
x4	.27	.73

Step 2:

$$j=1 \rightarrow c_1 = \frac{\sum_{i=1}^{n} w_{i1}^{2} x_i}{\sum_{i=1}^{n} w_{i1}^{2}}, \quad j=2 \rightarrow c_2 = \frac{\sum_{i=1}^{n} w_{i2}^{2} x_i}{\sum_{i=1}^{n} w_{i2}^{2}}$$

$(.4)^2+(.88)^2+(.41)^2+(.27)^2 = 1.18$

$(.6)^2+(.12)^2+(.59)^2+(.73)^2 = 1.25$

$$c_{11} = \frac{(.4)^2 \text{ x } 1 + (.88)2 \text{ x } 2 + (.41)2 \text{ x } 9 + (.27)2 \text{ x } 10}{1.18} = \frac{3.97}{1.18} = 3.38$$

$$c_{12} = \frac{(.4)^2 \text{ x } 2 + (.88)2 \text{ x } 3 + (.41)2 \text{ x } 4 + (.27)2 \text{ x } 1}{1.18} = \frac{3.39}{1.18} = 2.88$$

$centroid\ c_1 = [c_{11}, c_{12}] = [3.38, 2.88]$ $c2 = [c_{21}, c_{22}] = [7.02, 2.14]$

Fuzzy C-Means clustering (FCM) : Example

Input : Data :{(1,2) , (2,3) , (9.4), (10,1)}. , k=2 ,

Output: w_{ij} , $c_j, 1 \leq j \leq k$, $1 \leq i \leq n$

1. Initialization: Randomly select values for all w_{ij} *s.t.* $\sum_{j=1}^{k} w_{ij} = 1$
2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$;p=2.
3. Update the fuzzy-pseudo partition, w_{ij} : $w_{ij} = \frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$, if p = 2, then $w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$
4. Repeat step 2 and 3 until w_{ij} doesn't change (convergence).

Repeat

Data

	att1	att2
x1	1	2
x2	2	3
x3	9	4
x4	10	1

Determine new w_{ij} with new centroids

$$centroids\ c_1 = [\,c_{11}, c_{12}] = [3.38, 2.88]$$
$$c2 = [\,c_{21}, c_{22}] = [7.02, 2.14]$$

Step 3:

$$w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$$

$$w_{11} = \frac{(1/dist(x_1,c_1))}{(1/dist(x_1,c_1)+(1/dist(x_1,c_2))} = \frac{(1/2.54)}{(1/2.54)+(1/6.03)} = .7$$

$$w_{12} = \frac{(1/dist(x_1,c_2))}{(1/dist(x_1,c_1)+(1/dist(x_1,c_2))} = \frac{(1/6.03)}{(1/2.54)+(1/6.03)} = .3$$

Euclidean distance

	c1	c2
x1	2.54	6.03
x2	1.38	5.1
x3	5.73	2.71
x4	6.86	3.19

Updated weights

	c1	c2
x1	.7	.3
x2	.79	.21
x3	.32	.68
x4	.32	.68

FCM: Final Weights after a few iterations

Input : Data :{(1,2) , (2,3) , (9.4), (10,1)}. , k=2 ,

Output: w_{ij} , c_j, $1 \le j \le k$, $1 \le i \le n$

1. Initialization: Randomly select values for all w_{ij} *s.t.* $\sum_{j=1}^{k} w_{ij} = 1$
2. Compute centroids: $c_j = \frac{\sum_{i=1}^{n} w_{ij}^{p} x_i}{\sum_{i=1}^{n} w_{ij}^{p}}$;p=2.
3. Update the fuzzy-pseudo partition, w_{ij} : $w_{ij} = \frac{(1/dist(x_i,c_j))^{\frac{1}{p-1}}}{\sum_{q=1}^{k}(1/dist(x_i,c_q))^{\frac{1}{p-1}}}$, if p = 2, then $w_{ij} = \frac{(1/dist(x_i,c_j))}{\sum_{q=1}^{k}(1/dist(x_i,c_q))}$
4. Repeat step 2 and 3 until w_{ij} doesn't change (convergence).

Repeat

Data

	att1	att2
x1	1	2
x2	2	3
x3	9	4
x4	10	1

Finally, report the sum of squared deviations from the centroid:

$$\sum_{j=1}^{k}\sum_{i=1}^{n} w_{ij}^{p} dist(x_i, c_j)$$

$$P > 1$$

Final weights

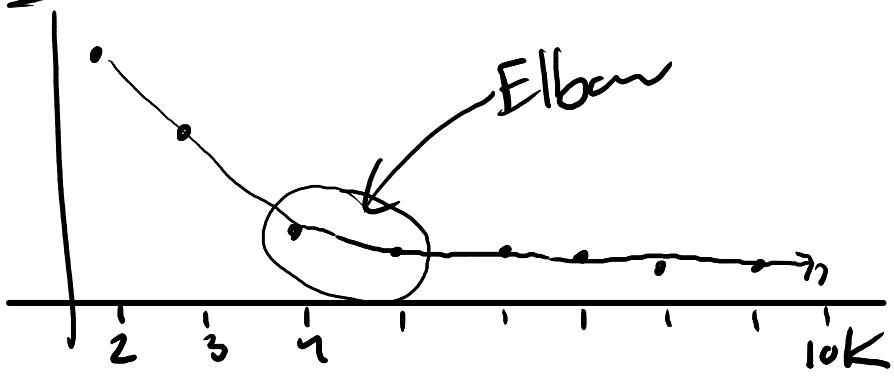
	c1	c2	Cluster
x1	.88	.12	c1
x2	.94	.06	c1
x3	.17	.83	c2
x4	.18	.82	c2

By tuning p and k values, Sum of the squared deviations from the centroid of clusters change.

Ideal case: when the Sum of the squared deviations from the cluster centroids are minimized.

#clusters?

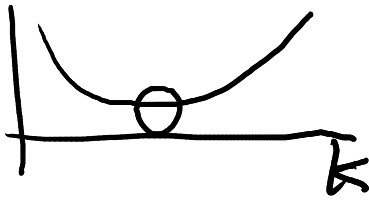
Elbow method



SSE + f(k)

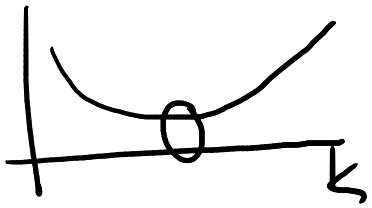
Akaike Info Criterion

AIC



Bayesian Info Criterion

BIC



Tibshirani

Performance?

- **SSE** } based on distances
- Silhouette score

↑ intrinsic

↓ extrinsic

- compare to domain knowledge
- visual inspection

Distance Metrics are key to clustering results

- Distance Measures
  - Define how the similarity of two elements (x,y) are calculated
  - Determines the shape of the clusters

- Distance Measure Techniques
  - Euclidean Distance, used in common clustering algorithms (k-Means etc)
  - Manhattan Distance
  - Correlation-based Distance : example is Pearson's correlation (sensitive to outliers), Spearman Correlation Distance (not sensitive to outliers)
  - Hamming Distance, used in information retrieval
  - Cosine Distance (vector space model)

Manhattan Distance

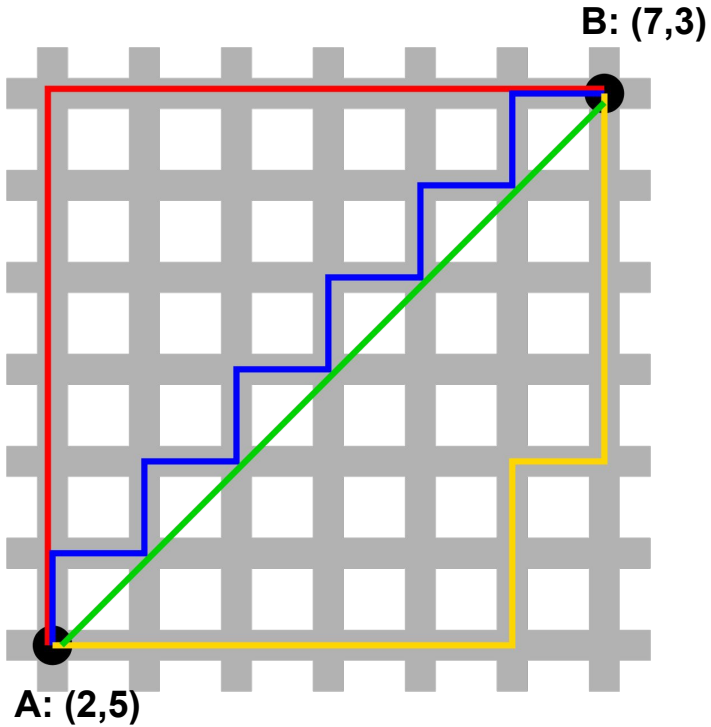
- Manhattan distance is the distance between two points in a grid based on a strictly horizontal and/or vertical path
  - Along grid lines, not diagonal distance
- The Manhattan distance is the sum of the absolute differences between the coordinates of the points along each dimension:

$$d_{\text{Manhattan}}(\mathbf{X}, \mathbf{Y}) = \| \mathbf{X} - \mathbf{Y} \|_1 = | x_1 - y_1 | + | x_2 - y_2 | + \cdots$$

L_1

- Example A and B:

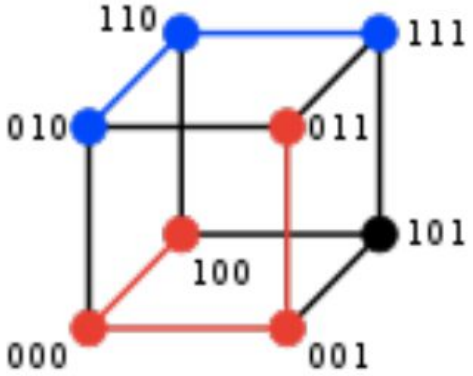
  Manhattan distance = |2 - 7| + |5 - 3| = 5 + 2 = 7



Hamming Distance

- The Hamming distance between **two strings** of equal length is the number of positions at which the corresponding symbols are different

Two example distances:
100→011 has distance 3;
010→111 has distance 2



3 bit binary cube for finding distance

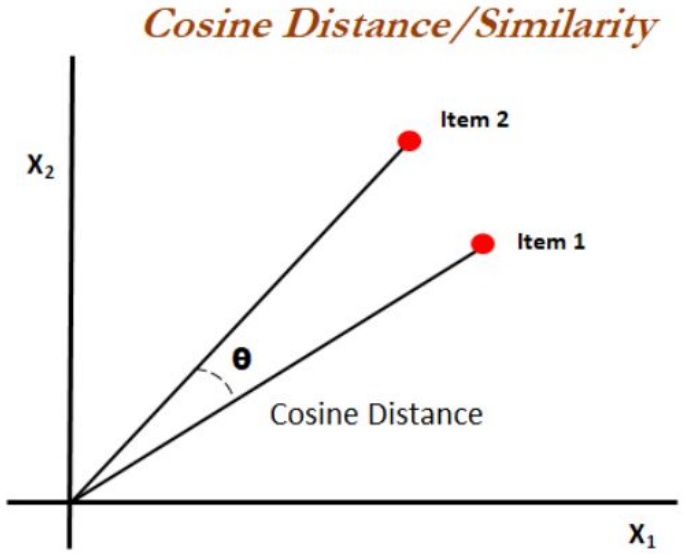
- The minimum distance between any two vertices is the Hamming distance between the two binary strings

Cosine distance

- Used in text and image processing applications → ath embeddings
- The cosine similarity is the cosine function of the angle between the two feature vectors in a multidimensional space:

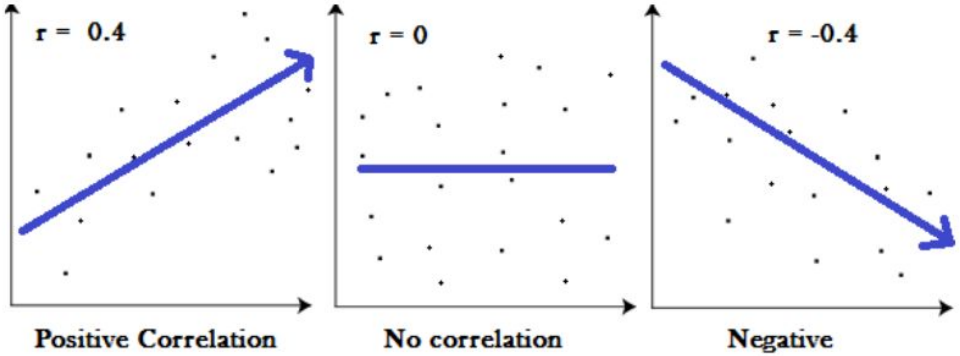
$$Cosine\ Distance = 1 - \cos(\theta)$$

$$\cos(\theta) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

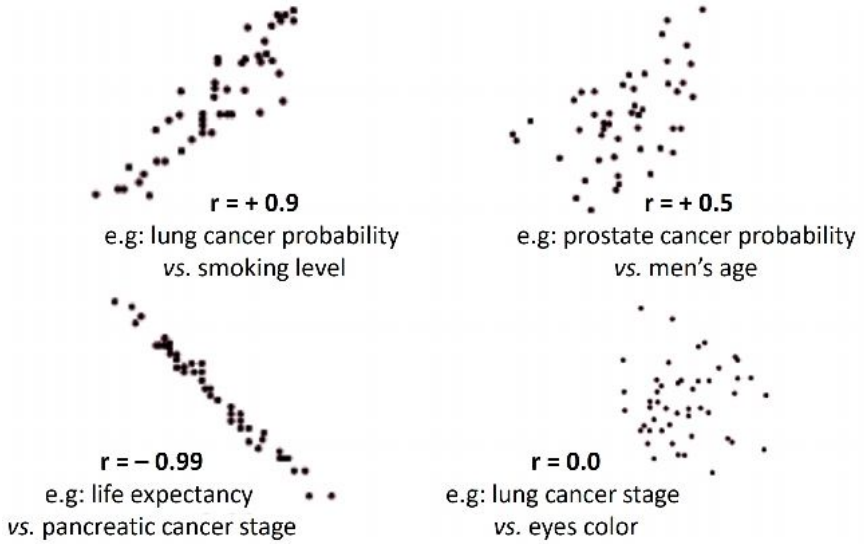


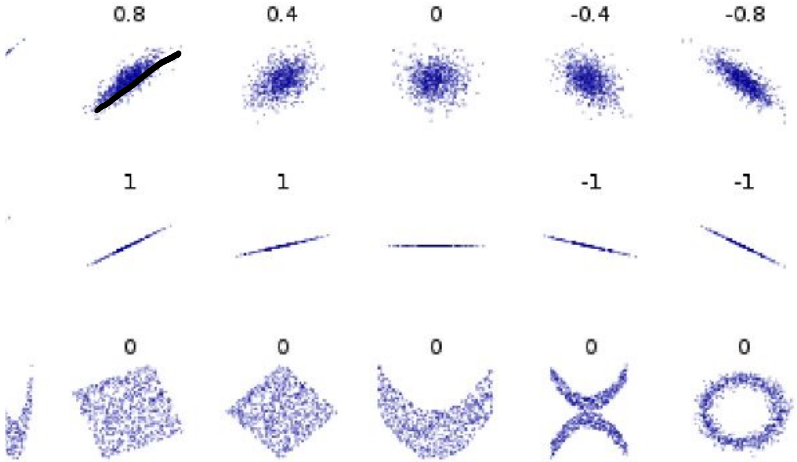
Pearson's Correlation coefficient

- The Pearson correlation coefficient (r) can be used in clustering as a similarity or distance measure between data points
- It is the measure of the **linear relationship** between two continuous variables.
- Values from -1 to +1
  - **+1** indicates a perfect positive linear correlation
  - **-1** indicates a perfect negative linear correlation
  - **0** indicates no linear correlation between the variables



Pearson's Correlation coefficient measures linear relations





Pearson's Correlation coefficient measures linear relations

Equation:

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}.$$

n: number of samples

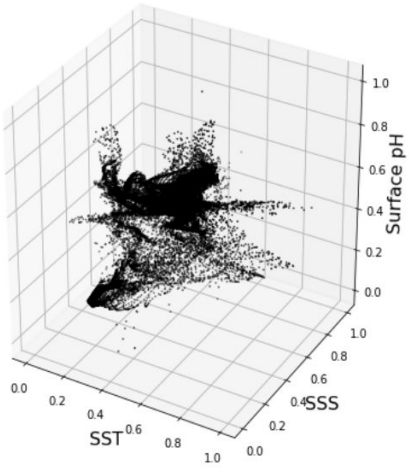
X: [10, 20, 30, 40, 50] , Y: [15, 25, 35, 45, 55]

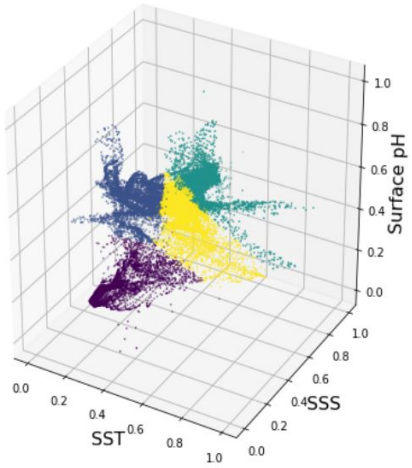
1. Calculate the mean of X ($\bar{X}$) and the mean of Y ($\bar{Y}$).
   - $\bar{X}$ = (10 + 20 + 30 + 40 + 50) / 5 = 30
   - $\bar{Y}$ = (15 + 25 + 35 + 45 + 55) / 5 = 35
2. Calculate the deviations from the mean for X (Xdev) and Y (Ydev) :
   - Xdev: [-20, -10, 0, 10, 20]
   - Ydev: [-20, -10, 0, 10, 20]
3. Calculate the sum of the products of the deviations (Xdev * Ydev):
   - Σ(Xdev * Ydev) = (-20 * -20) + (-10 * -10) + (0 * 0) + (10 * 10) + (20 * 20) = 900
4. Calculate the sum of the squared deviations for X (Σ(Xdev^2)) and Y (Σ(Ydev^2)):
   - Σ(Xdev^2) = 400 + 100 + 0 + 100 + 400 = 1000
   - Σ(Ydev^2) = 400 + 100 + 0 + 100 + 400 = 1000
5. Calculate the square root of the product of the sums of squared deviations:
   - $\sqrt{(\Sigma(\text{Xdev}^2) * \Sigma(\text{Ydev}^2))} = \sqrt{(1000 * 1000)} = 1000$
6. Calculate the Pearson correlation coefficient (r) :
   - r = $\Sigma(\text{Xdev} * \text{Ydev}) / \sqrt{(\Sigma(\text{Xdev}^2) * \Sigma(\text{Ydev}^2))}$ = 900 / 1000 = 0.9

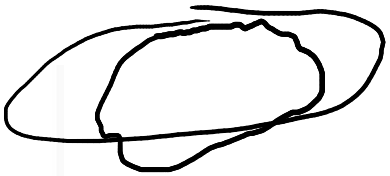
k-Means and Fuzzy c-means make strong assumptions

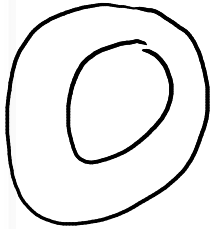
Gaussian Elliptical

Blob-shaped

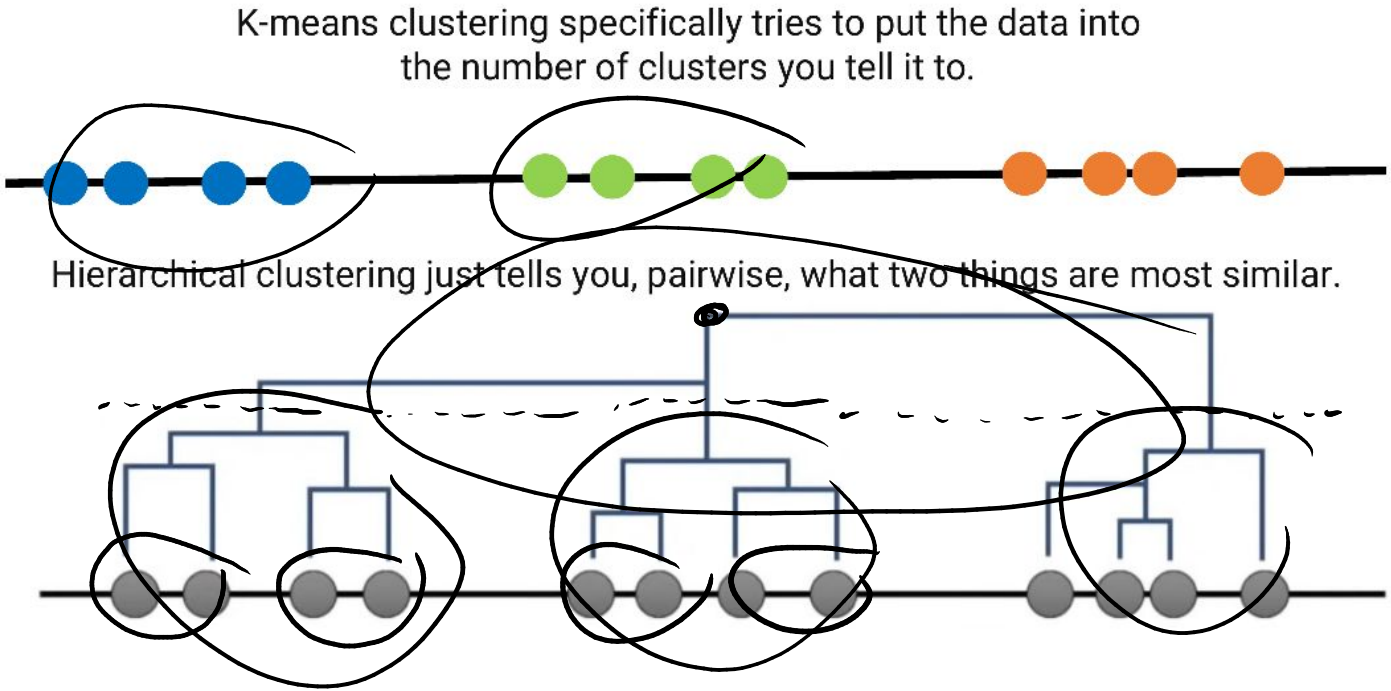








Partitional (e.g. k-Means) vs hierarchical



Hierarchical clustering

Hierarchical clustering is one of the popular and easy to understand clustering technique. This clustering technique is divided into two types:

1. Agglomerative: Initially all data points are one cluster which we merge iteratively
2. Divisive: Initially there is only one cluster which we subdivide

Linkage criteria, how distances are measured, are key here also

Hierarchical Clustering: Distance

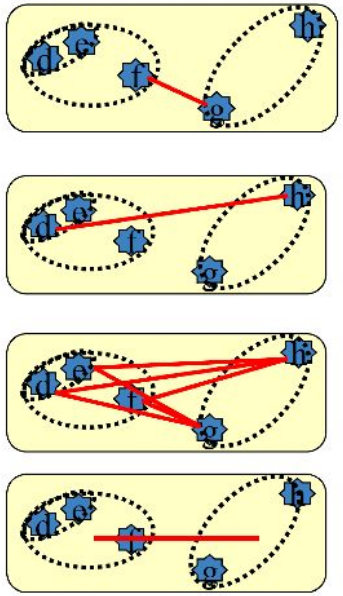
We may define as “distance” the:

• Minimum distance between elements of each cluster (single-linkage clustering)

•Maximum distance between elements of each cluster (complete-linkage clustering)

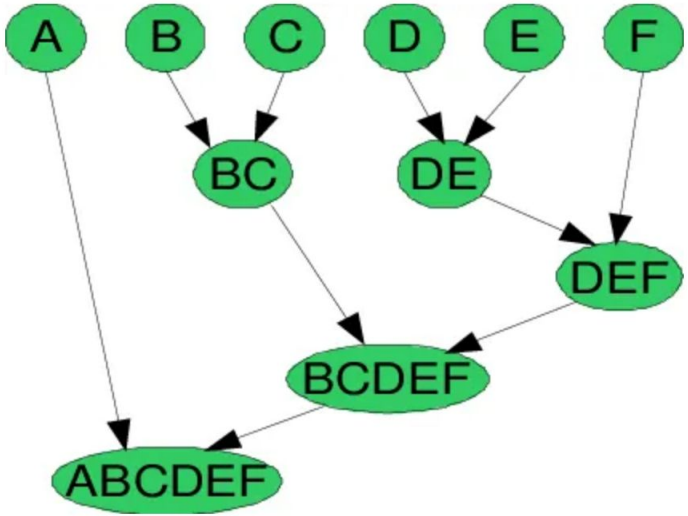
•Mean distance between all elements in each cluster (average-linkage clustering)

•Distance between the centroids of each cluster (centroid-linkage clustering)



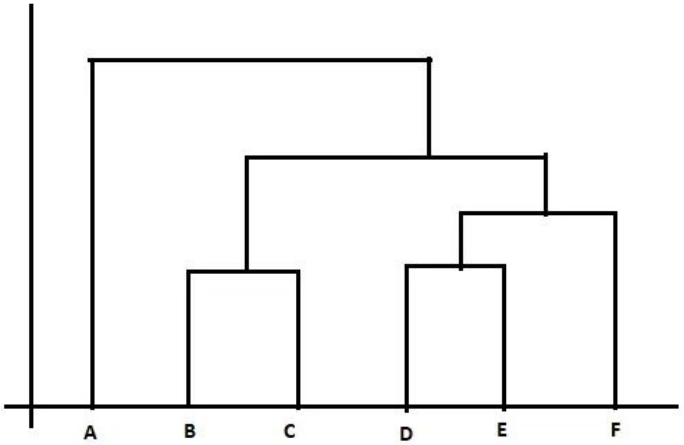
@ IliasTagkopoulos

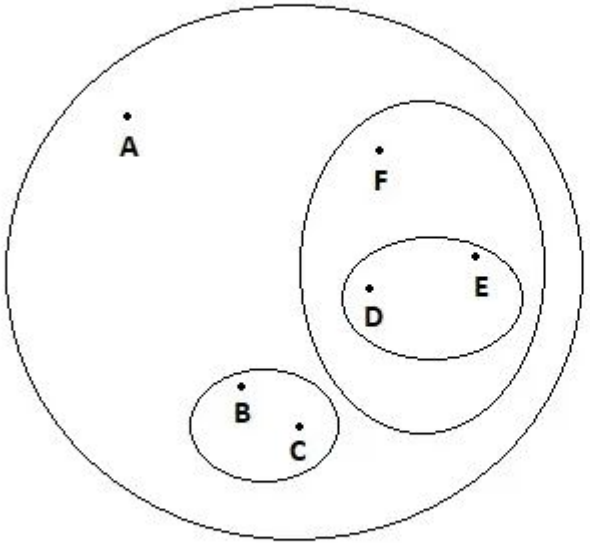
Hierarchical clustering: Agglomerative



Agglomerative Hierarchical Clustering Technique

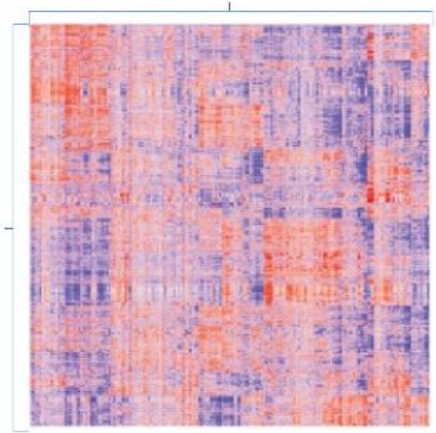
Hierarchical Clustering: Dendrograms





Example: Hierarchical gene-expression analysis

The columns represent different samples



The rows represent measurements from different genes.

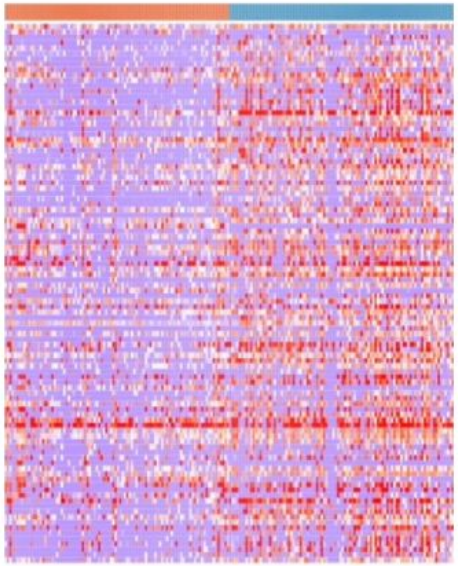
Hierarchical clustering orders the rows and/or the columns based on similarity

This makes it easy to see correlations in the data

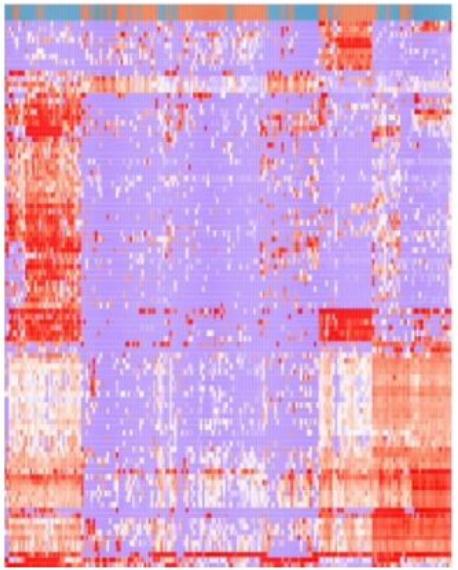
Red usually stands for high expression of gene.
Blue/Purple usually stands for low expression of gene.

Heatmap with hierarchical clustering

The heatmap **without** hierarchical clustering

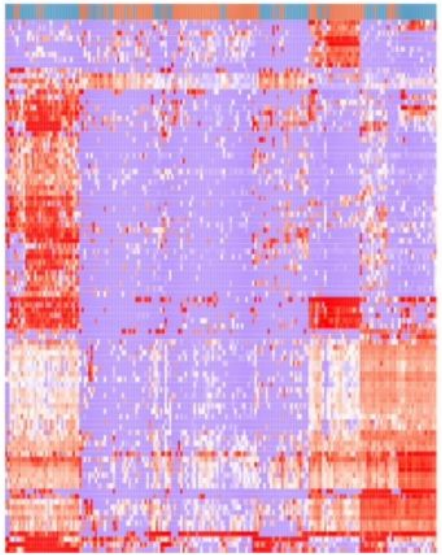


The heatmap **with** hierarchical clustering

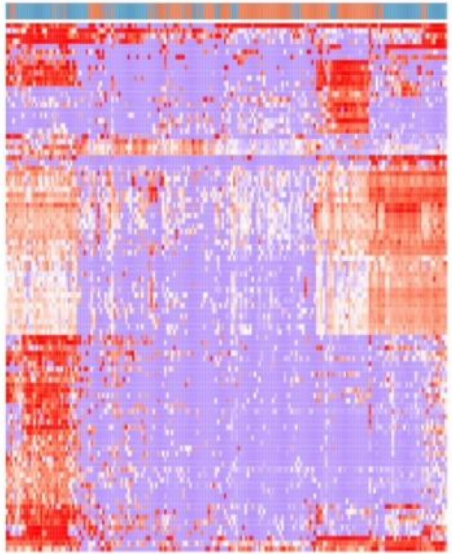


The rows and columns of the heatmap are reordered to place similar data points closer together

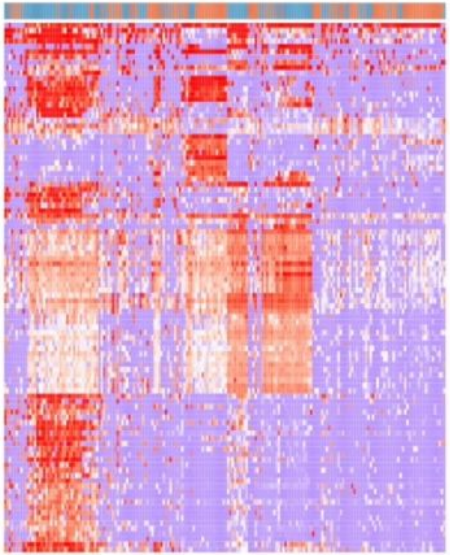
Hierarchical gene-expression analysis: Distance metric



Here's a heatmap that compares the **furthest** points in the clusters.



Here's a heatmap that compares the **average** points in the clusters.



Here's a heatmap that compares the **closest** points in the clusters.

Density based clustering

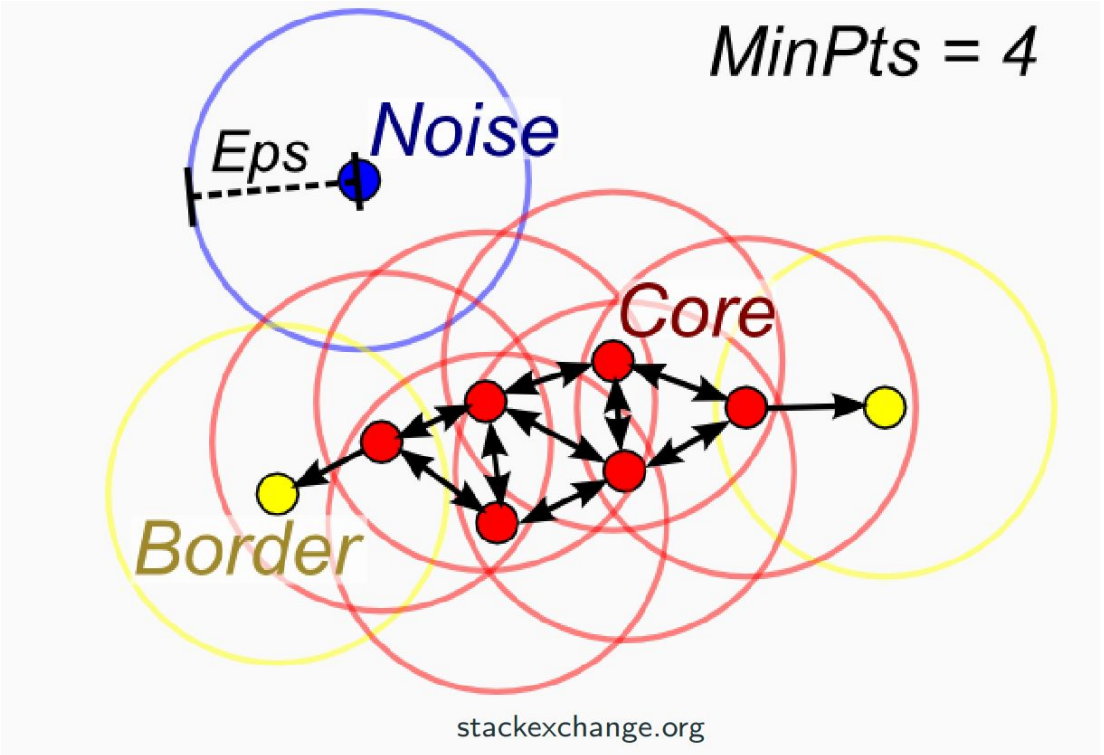
- Most popular: Density-based spatial clustering of applications with noise (DBSCAN)
- Non-parametric algorithm (no ‘k to choose’):
  - Given a set of points in some space, it groups points that are closely packed together
- Marking points as outliers that lie alone in low-density regions
  - Nearest neighbors are ‘too far’ away

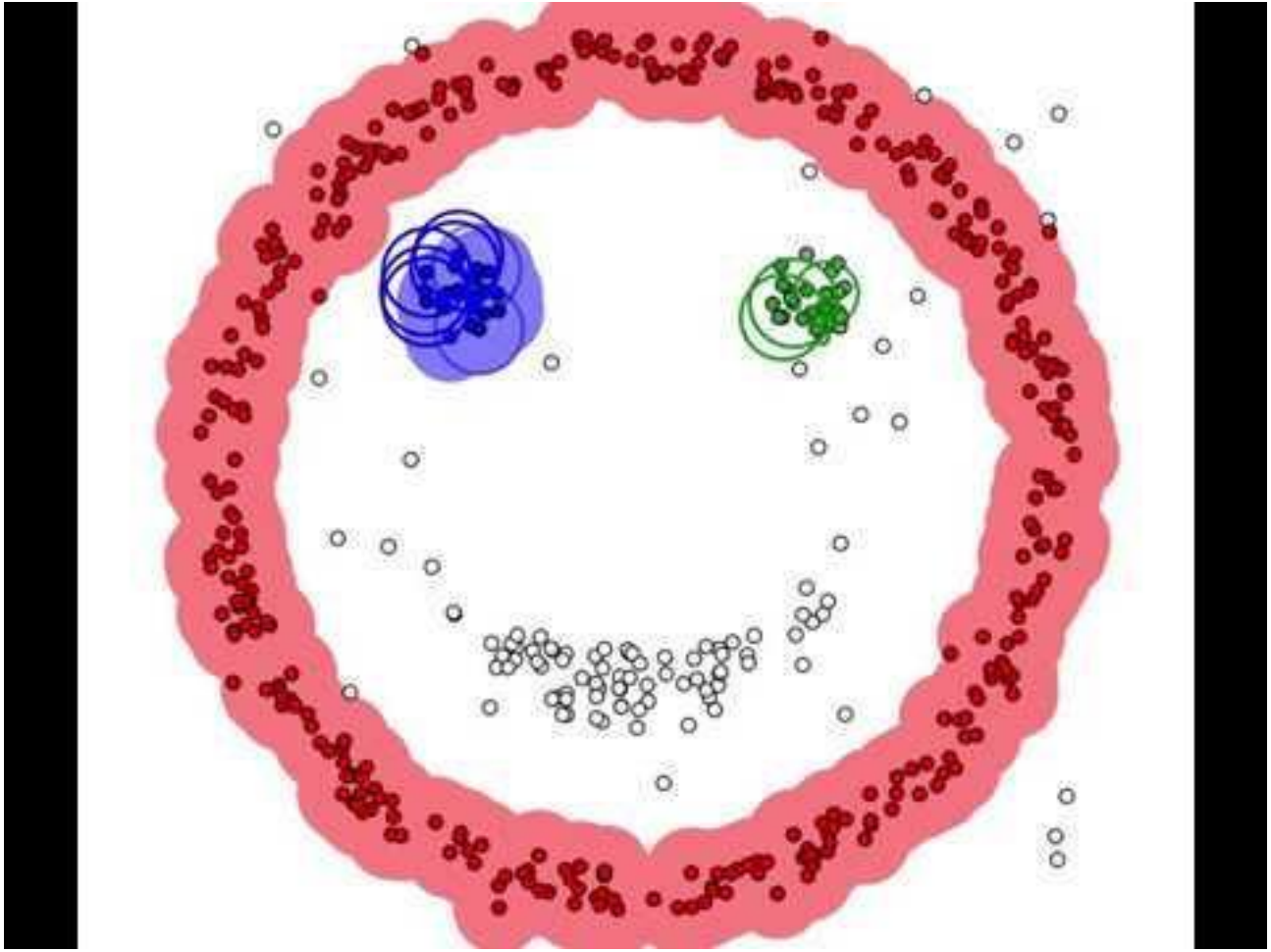
Density-based spatial clustering of applications with noise (DBSCAN)

- Set: Eps and minimum points (MinPts)

```
DBSCAN(dataset, eps, MinPts){
# cluster index
C = 1
for each unvisited point p in dataset {
         mark p as visited
         # find neighbors
         Neighbors N = find the neighboring points of p

         if |N|>=MinPts:
             N = N U N'
             if p' is not a member of any cluster:
                 add p' to cluster C
}
```





Choosing a clustering algorithm appropriate for the data…