Clustering

Supervised vs. Unsupervised Learning

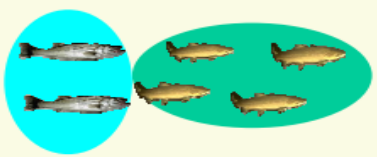
- Up to now we considered ***supervised learning*** scenario, where we are given
  1. samples $x_1, \ldots, x_n$
  2. class labels for all samples $x_1, \ldots, x_n$
  - This is also called learning with teacher, since correct answer (the true class) is provided

- In the next few lectures we consider ***unsupervised learning*** scenario, where we are only given
  1. samples $x_1, \ldots, x_n$
  - This is also called learning without teacher, since correct answer is not provided
  - do not split data into training and test sets

Unsupervised Learning

- Data is *not labeled*

1. Parametric Approach
   - assume parametric distribution of data
   - estimate parameters of this distribution
   - much “harder” than supervised case
- NonParametric Approach
   - group the data into ***clusters***, each cluster (hopefully) says something about categories (classes) present in the data



a lot is known ”easier”

little is known “harder”

Why Unsupervised Learning?

- Unsupervised learning is harder
  - How do we know if results are meaningful? No answer labels are available.
    - Let the expert look at the results (external evaluation)
    - Define an objective function on clustering (internal evaluation)
- We nevertheless need it because
  1. Labeling large datasets is very costly (speech recognition)
     - sometimes can label only a few examples by hand
  2. May have no idea what/how many classes there are (data mining)
  3. May want to use clustering to gain some insight into the structure of the data before designing a classifier
     - Clustering as data description

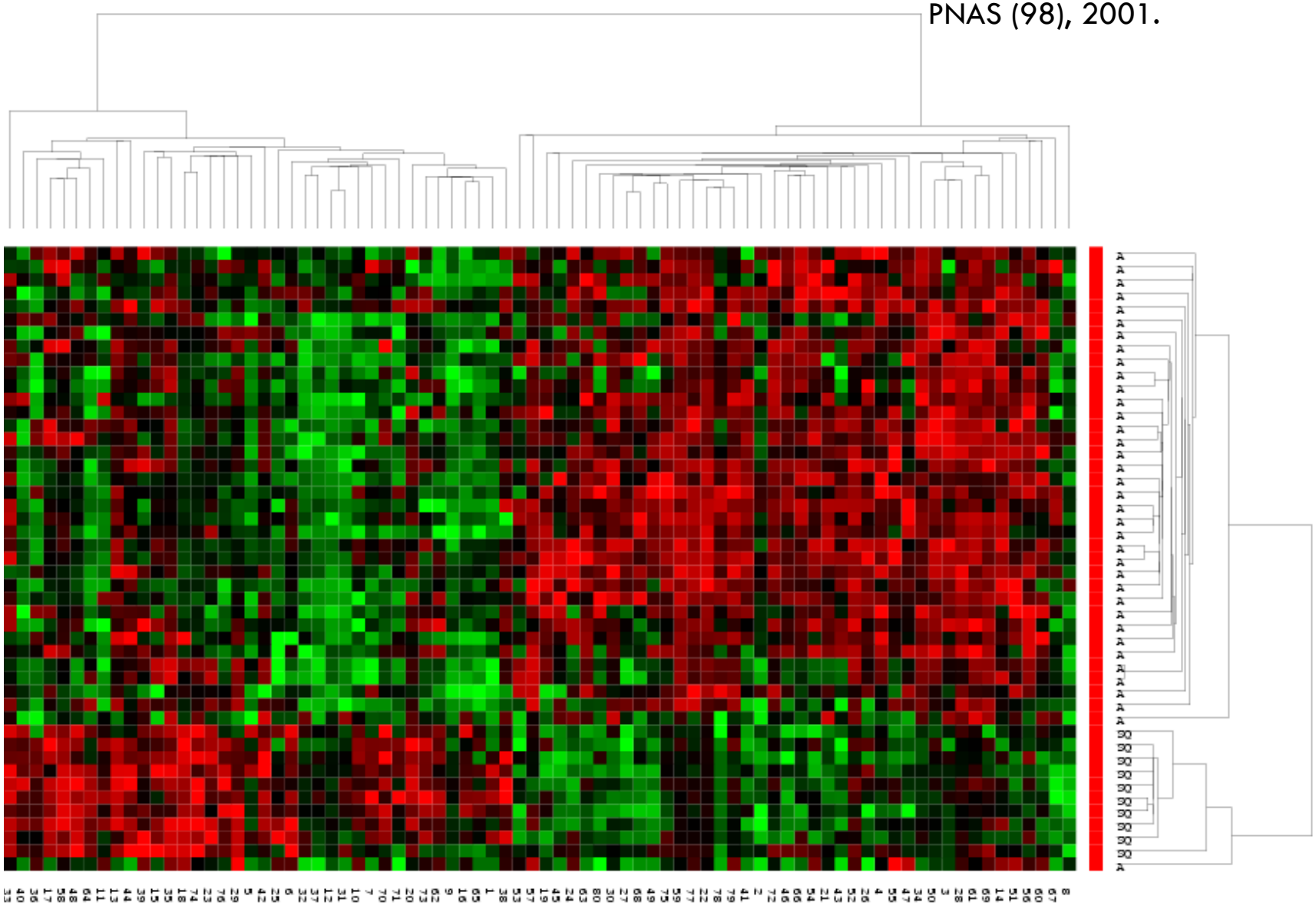
Clustering

Clustering: the process of grouping a set of objects into classes of similar objects

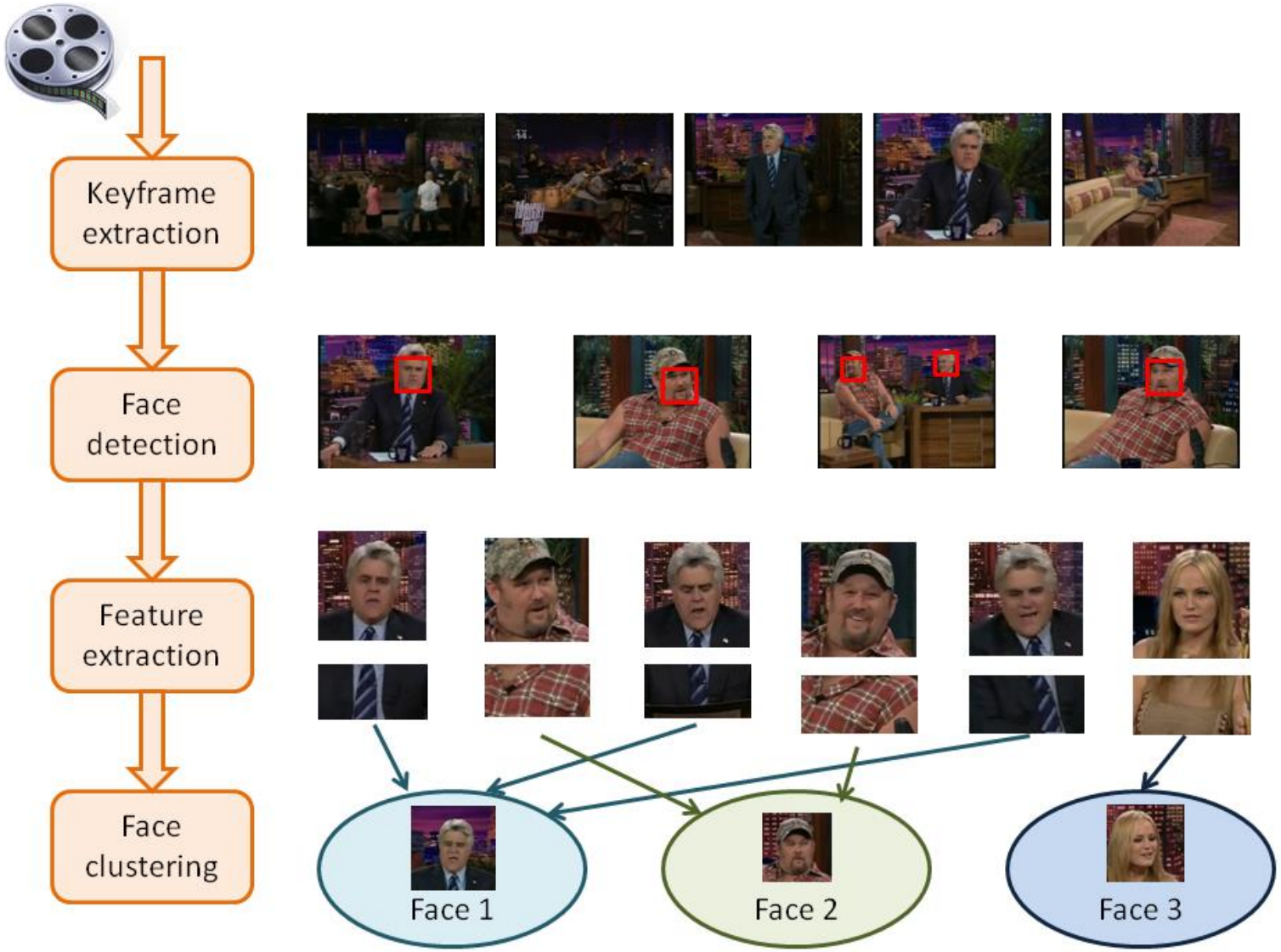
Applications?

Gene expression data

Data from Garber et al.
PNAS (98), 2001.

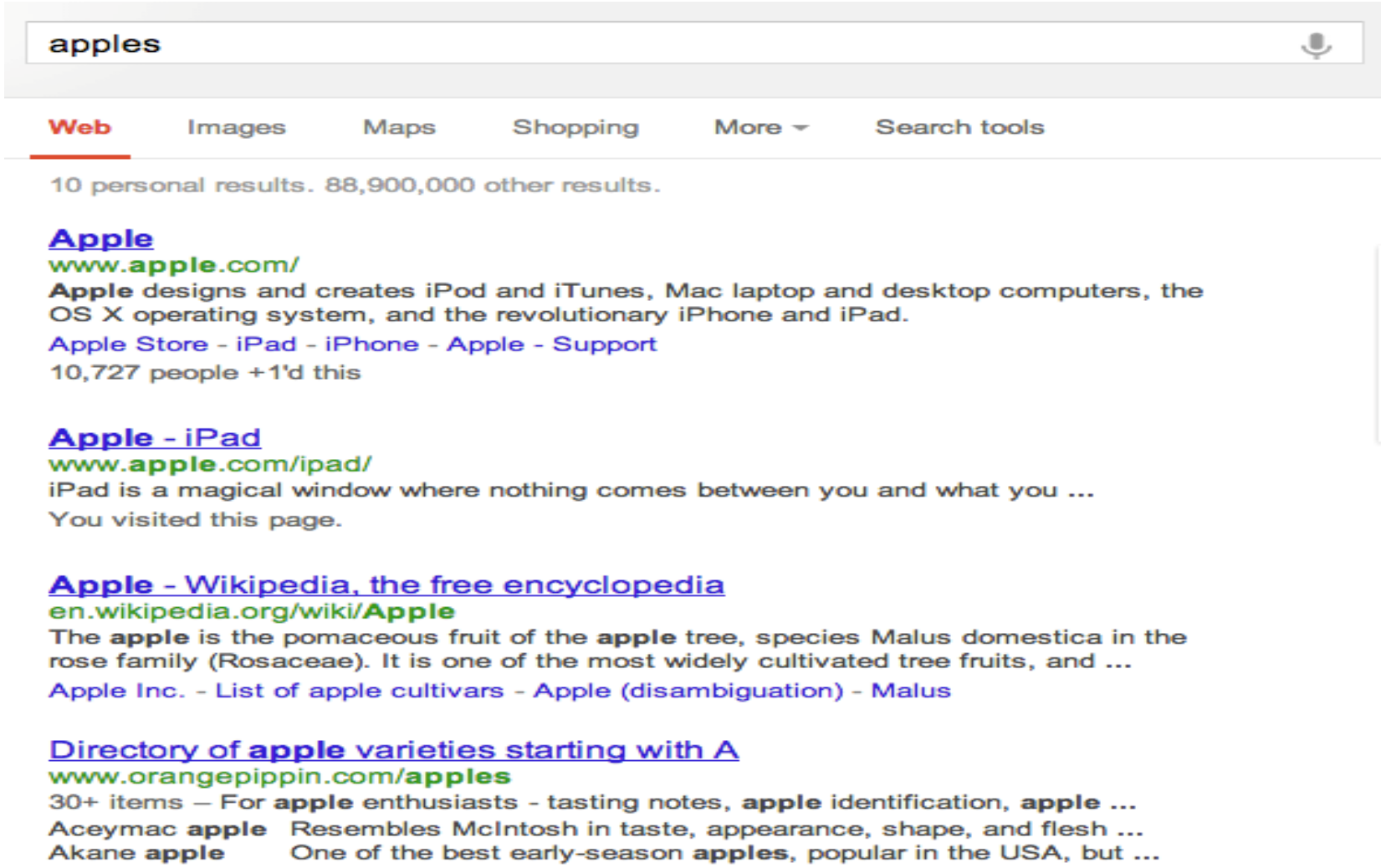


Face Clustering



Face clustering

Search result clustering



Google News

Google

News

Top Stories

- Iran
- Xbox One
- Tarun Tejpal
- Manny Pacquiáo
- Ukraine
- Kabul
- New England Patriots
- Latvia
- Derrick Rose
- Doctor Who

Xbox One

E! Online

Console Wars 2013: Microsoft's Xbox One vs. Sony's PlayStation 4

E! Online - 1 hour ago

The future is now! Last week, Sony released its next generation console, PlayStation 4. This weekend, Microsoft drops the much touted all-in-one media device, Xbox One. We've been geeking out over the two new systems, and compiling a report on the new ...

Xbox One sales exceed one million in first 24 hours Joystiq - by David Hinkle

Xbox One vs. PS4: A Guide to Making the Toughest Gaming Decision in Years ABC News - by Joanna Stern

Related Microsoft »

See realtime coverage

Wall Street Journal

YouTube

YouTube

Washingto...

RedOrbit

Guardian ...

The Guardian

Xbox One and Microsoft websites marred by problems on launch day

The Guardian
9 hours ago

Written by Jemima Kiss

Microsoft's Xbox One launch was marred by problems with its online services early on Friday which took down the official website Xbox.

Consumers line up for Xbox One

USA TODAY - Nov 23, 2013

Eager video game players lined up at stores across the country awaiting the arrival of Microsoft's Xbox One, a week to the day after rival Sony introduced its PlayStation 4. The console, available for sale tonight at 12:01 a.m.

Product Re...

Here are all the Xbox One voice commands

Polygon
9 hours ago

Written by Megan Farokhmanesh

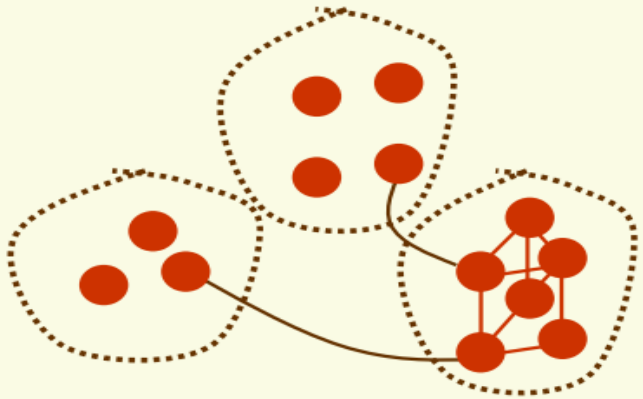
Microsoft posted a guide to Xbox One voice commands, including how to navigate menus, control volume and multitask, on its Tumblr.

Clustering for Data Understanding and Applications

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults
- Climate: understanding earth climate, find patterns of atmospheric and ocean
- Economic Science: market resarch

Clustering

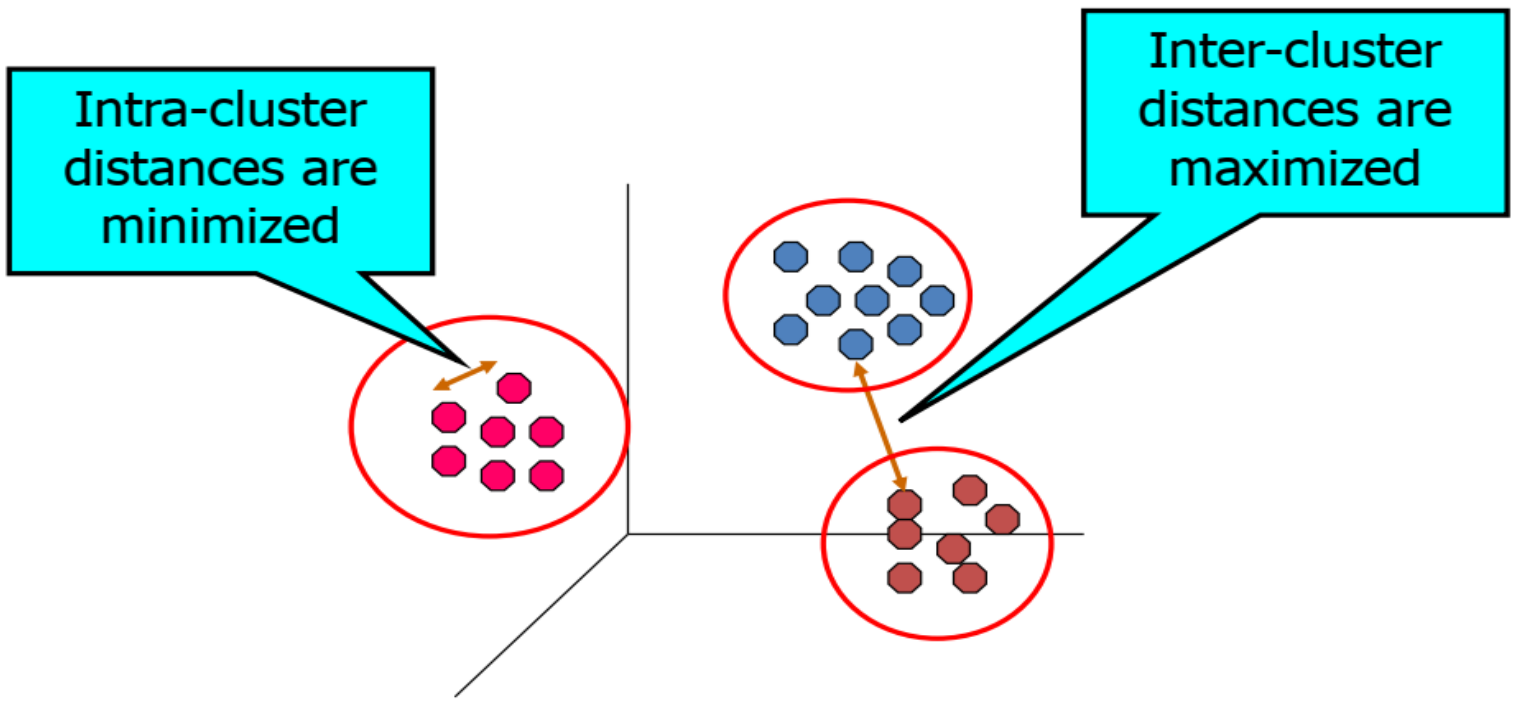
- Seek “natural” clusters in the data



- What is a good clustering?
  - internal (within the cluster) distances should be small
  - external (intra-cluster) should be large
- Clustering is a way to discover new categories (classes)

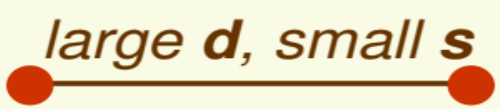
Clustering

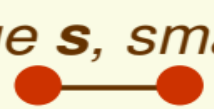
- Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



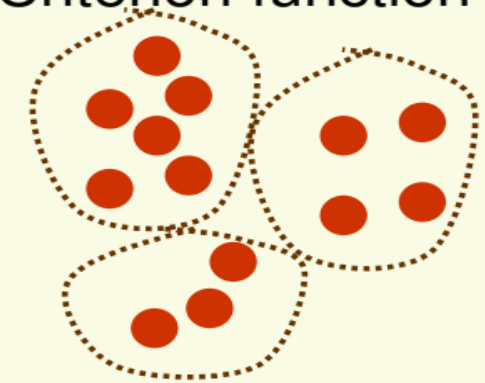
What we Need for Clustering

1. Proximity measure, *either*
   - similarity measure $s(x_i, x_k)$: large if x_i, x_k are similar
   - dissimilarity(or distance) measure $d(x_i, x_k)$: small if x_i, x_k are similar

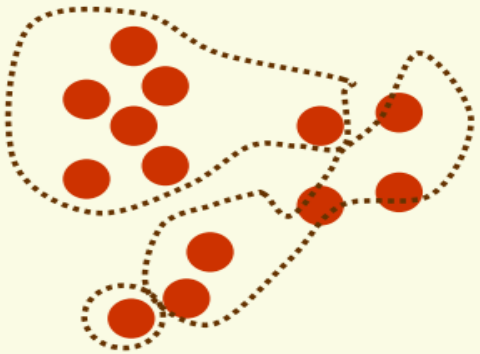




2. Criterion function to evaluate a clustering



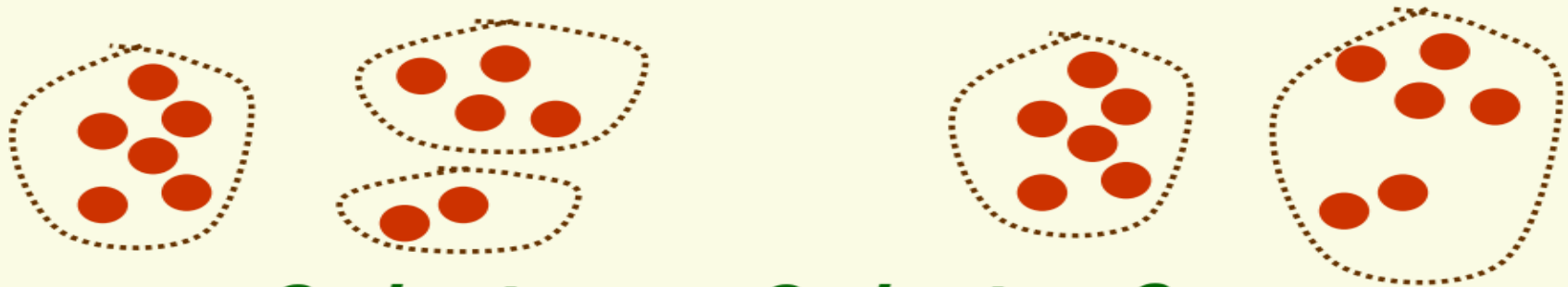
good clustering



bad clustering

3. Algorithm to compute clustering
   - For example, by optimizing the criterion function

How Many Clusters?

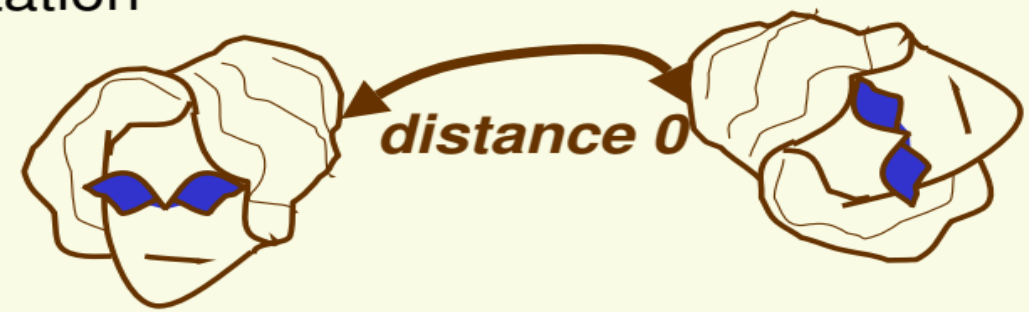


3 clusters or 2 clusters?

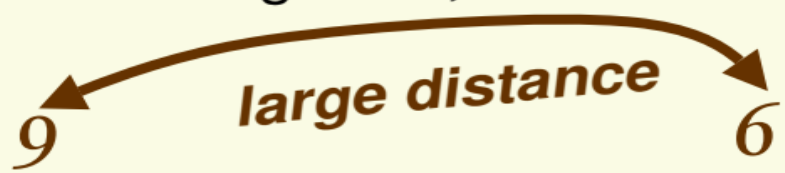
- Possible approaches
  1. fix the number of clusters to ***k***
  2. find the best clustering according to the criterion function (number of clusters may vary)

Proximity Measures

- good proximity measure is VERY application dependent
  - Clusters should be invariant under the transformations "natural" to the problem
  - For example for object recognition, should have invariance to rotation



  - For character recognition, no invariance to rotation

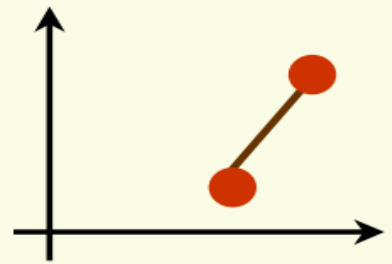


Distance (dissimilarity) Measures

- Euclidean distance

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{d} \left(x_i^{(k)} - x_j^{(k)}\right)^2}$$

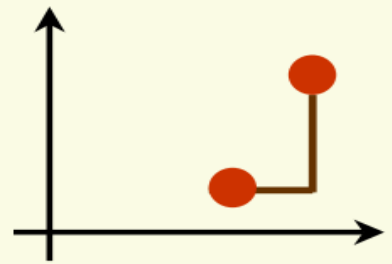
  - translation invariant



- Manhattan (city block) distance

$$d(x_i, x_j) = \sum_{k=1}^{d} \left| x_i^{(k)} - x_j^{(k)} \right|$$

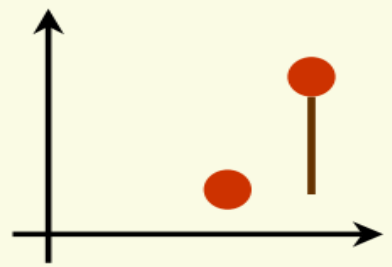
  - approximation to Euclidean distance, cheaper to compute



- Chebyshev distance

$$d(x_i, x_j) = \max_{1 \le k \le d} \left| x_i^{(k)} - x_j^{(k)} \right|$$

  - approximation to Euclidean distance, cheapest to compute



The most popular distance measure is **Euclidean distance** (i.e., straight line or "as the crow flies"). Let $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ be two objects described by p numeric attributes. The Euclidean distance between objects i and j is defined as

$$d(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}. \tag{2.16}$$

Another well-known measure is the **Manhattan (or city block) distance**, named so because it is the distance in blocks between any two points in a city (such as 2 blocks down and 3 blocks over for a total of 5 blocks). It is defined as

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}|. \tag{2.17}$$

Both the Euclidean and the Manhattan distance satisfy the following mathematical properties:

Non-negativity: $d(i, j) \geq 0$: Distance is a non-negative number.

Identity of indiscernibles: $d(i, i) = 0$: The distance of an object to itself is 0.

Symmetry: $d(i, j) = d(j, i)$: Distance is a symmetric function.

Triangle inequality: $d(i, j) \leq d(i, k) + d(k, j)$: Going directly from object i to object j in space is no more than making a detour over any other object k.

Example 2.19 **Euclidean distance and Manhattan distance.** Let $x_1 = (1, 2)$ and $x_2 = (3, 5)$ represent two objects as shown in Figure 2.23. The Euclidean distance between the two is $\sqrt{2^2 + 3^2} = 3.61$. The Manhattan distance between the two is $2 + 3 = 5$. ∎

Minkowski distance is a generalization of the Euclidean and Manhattan distances. It is defined as

$$d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h}, \tag{2.18}$$

where h is a real number such that $h \geq 1$. (Such a distance is also called L_p **norm** in some literature, where the symbol p refers to our notation of h. We have kept p as the number of attributes to be consistent with the rest of this chapter.) It represents the Manhattan distance when $h = 1$ (i.e., L_1 norm) and Euclidean distance when $h = 2$ (i.e., L_2 norm).

The **supremum distance** (also referred to as L_{max}, L_∞ **norm** and as the **Chebyshev distance**) is a generalization of the Minkowski distance for $h \to \infty$. To compute it, we find the attribute f that gives the maximum difference in values between the two objects. This difference is the supremum distance, defined more formally as:

$$d(i,j) = \lim_{h\to\infty} \left(\sum_{f=1}^{p} |x_{if} - x_{jf}|^h \right)^{\frac{1}{h}} = \max_{f}^{p} |x_{if} - x_{jf}|. \tag{2.19}$$

The L^∞ norm is also known as the *uniform norm.*

Example 2.20 **Supremum distance.** Let's use the same two objects, $x_1 = (1, 2)$ and $x_2 = (3, 5)$, as in Figure 2.23. The second attribute gives the greatest difference between values for the objects, which is $5 - 2 = 3$. This is the supremum distance between both objects. ∎

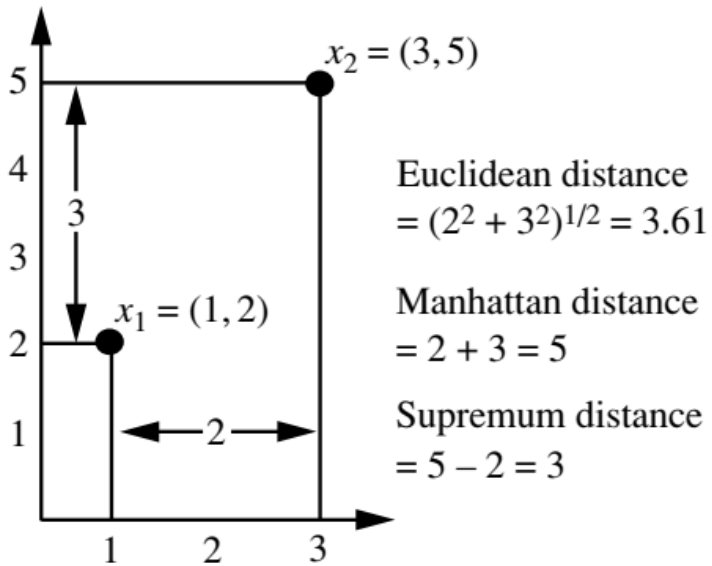
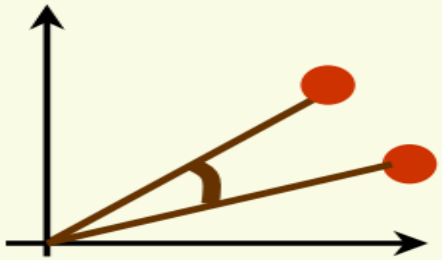


Figure 2.23 Euclidean, Manhattan, and supremum distances between two objects.

Similarity Measures

- Cosine similarity:

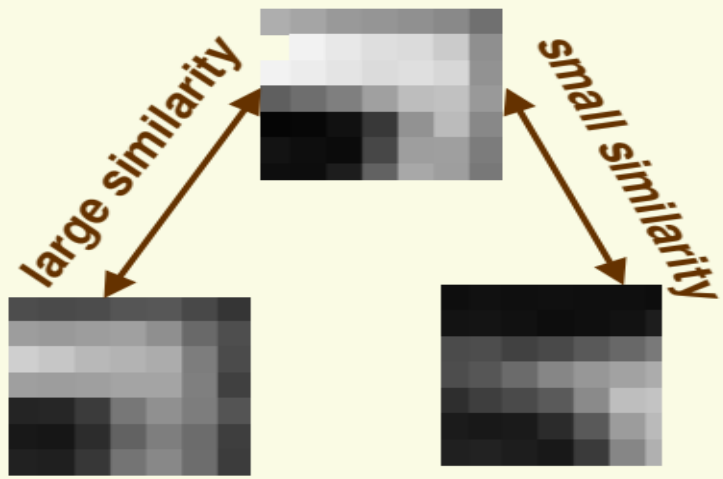
$$s(x_i, x_j) = \frac{x_i^T x_j}{\| x_i \| \, \| x_j \|}$$



  - the smaller the angle, the larger the similarity
  - scale invariant measure
  - popular in text retrieval

- Correlation coefficient
  - popular in image processing

$$s(x_i, x_j) = \frac{\sum_{k=1}^{d} (x_i^{(k)} - \bar{x}_i)(x_i^{(k)} - \bar{x}_i)}{\left[\sum_{k=1}^{d} (x_i^{(k)} - \bar{x}_i)^2 \sum_{k=1}^{d} (x_j^{(k)} - \bar{x}_j)^2 \right]^{1/2}}$$



Cosine Similarity

$$sim(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{||\boldsymbol{x}||||\boldsymbol{y}||}, \tag{2.23}$$

where $||\boldsymbol{x}||$ is the Euclidean norm of vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_p)$, defined as $\sqrt{x_1^2 + x_2^2 + \cdots + x_p^2}$. Conceptually, it is the length of the vector. Similarly, $||\boldsymbol{y}||$ is the Euclidean norm of vector $\boldsymbol{y}$. The measure computes the cosine of the angle between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. A cosine value of 0 means that the two vectors are at 90 degrees to each other (orthogonal) and have no match. The closer the cosine value to 1, the smaller the angle and the greater the match between vectors. Note that because the cosine similarity measure does not obey all of the properties of Section 2.4.4 defining metric measures, it is referred to as a *nonmetric measure*.

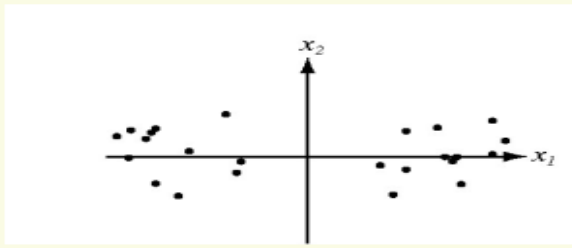
Example 2.23 **Cosine similarity between two term-frequency vectors.** Suppose that $\boldsymbol{x}$ and $\boldsymbol{y}$ are the first two term-frequency vectors in Table 2.5. That is, $\boldsymbol{x} = (5,0,3,0,2,0,0,2,0,0)$ and $\boldsymbol{y} = (3,0,2,0,1,1,0,1,0,1)$. How similar are $\boldsymbol{x}$ and $\boldsymbol{y}$? Using Eq. (2.23) to compute the cosine similarity between the two vectors, we get:

$$\begin{aligned}
\boldsymbol{x}^t \cdot \boldsymbol{y} &= 5 \times 3 + 0 \times 0 + 3 \times 2 + 0 \times 0 + 2 \times 1 + 0 \times 1 + 0 \times 0 + 2 \times 1 \\
&\quad + 0 \times 0 + 0 \times 1 = 25 \\
||\boldsymbol{x}|| &= \sqrt{5^2 + 0^2 + 3^2 + 0^2 + 2^2 + 0^2 + 0^2 + 2^2 + 0^2 + 0^2} = 6.48 \\
||\boldsymbol{y}|| &= \sqrt{3^2 + 0^2 + 2^2 + 0^2 + 1^2 + 1^2 + 0^2 + 1^2 + 0^2 + 1^2} = 4.12 \\
sim(\boldsymbol{x}, \boldsymbol{y}) &= 0.94
\end{aligned}$$

Therefore, if we were using the cosine similarity measure to compare these documents, they would be considered quite similar. ■

Feature Scale

- old problem: how to choose appropriate relative scale for features?
  - [length (in meters or cms?), weight(in in grams or kgs?)]
  - In supervised learning, can normalize to zero mean unit variance with no problems
  - in clustering this is more problematic, ***if variance in data is due to cluster presence, then normalizing features is not a good thing***

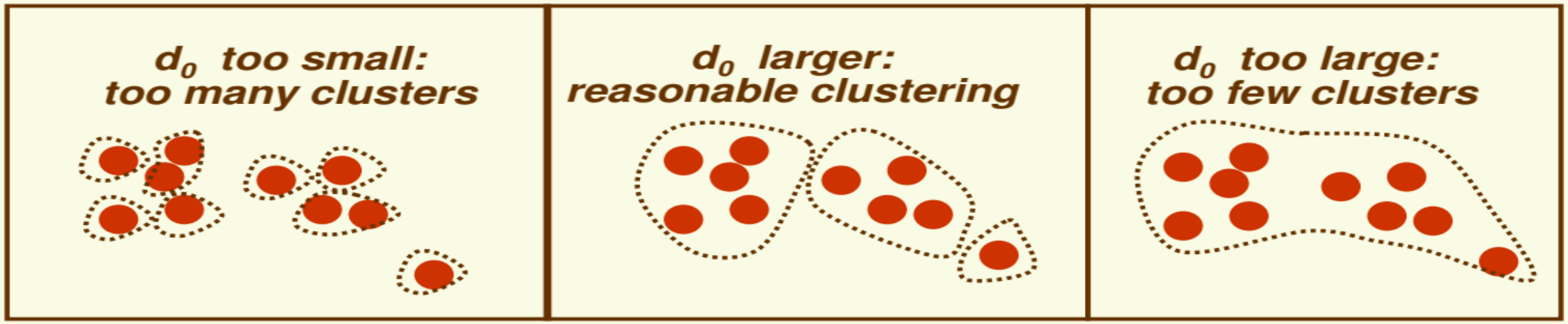


before normalization

after normalization

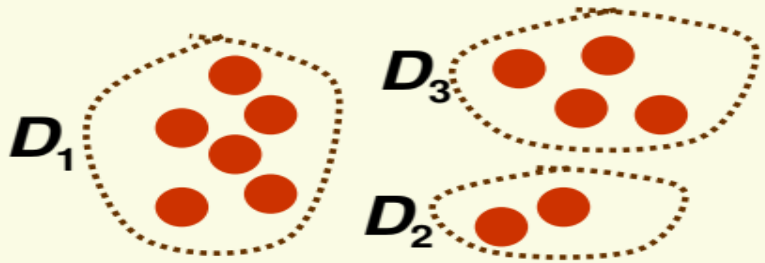
Simplest Clustering Algorithm

- Having defined a proximity function, can develop a simple clustering algorithm
  - go over all sample pairs, and put them in the same cluster if the distance between them is less then some threshold distance d_0 (or if similarity is larger than s_0)
  - Pros: simple to understand and implement
  - Cons: very dependent on d_0 (or s_0), automatic choice of d_0 (or s_0)is not an easily solved issue



Criterion Functions for Clustering

- Have samples $x_1,\ldots,x_n$
- Suppose partitioned samples into c subsets $D_1,\ldots,D_c$



- There are approximately $c^n/c!$ distinct partitions
- Can define a criterion function $J(D_1,\ldots,D_c)$ which measures the quality of a partitioning $D_1,\ldots,D_c$
- Then the clustering problem is a well defined problem
  - the optimal clustering is the partition which optimizes the criterion function

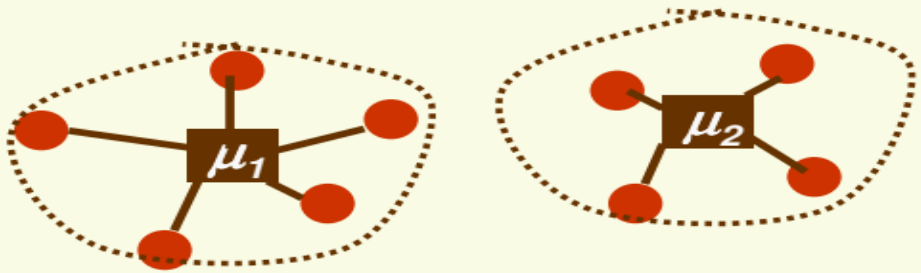
SSE Criterion Function

- Let n_i be the number of samples in D_i, and define the mean of samples in in D_i

$$\mu_i = \frac{1}{n_i} \sum_{x \in D_i} x$$

- Then the sum-of-squared errors criterion function (to minimize) is:

$$J_{SSE} = \sum_{i=1}^{c} \sum_{x \in D_i} || x - \mu_i ||^2$$



- Note that the number of clusters, c, is fixed

K-Means clustering

- K-means (MacQueen, 1967) is a partitional clustering algorithm
- Let the set of data points D be $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$,

  where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a vector in $X \subseteq R^r$, and r is the number of dimensions.
- The k-means algorithm partitions the given data into k clusters:
  - Each cluster has a cluster **center**, called **centroid**.
  - k is specified by the user

K-means algorithm

- Given *k*, the *k-means* algorithm works as follows:
  1. Choose *k* (random) data points (seeds) to be the initial centroids, cluster centers
  2. Assign each data point to the closest centroid
  3. Re-compute the centroids using the current cluster memberships
  4. If a convergence criterion is not met, repeat steps 2 and 3

K-means convergence (stopping) criterion

- no (or minimum) re-assignments of data points to different clusters, *or*
- no (or minimum) change of centroids, *or*
- minimum decrease in the **sum of squared error** (SSE),

$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} d(\mathbf{x}, \mathbf{m}_j)^2$$

  - C_j is the jth cluster,
  - $\mathbf{m}_j$ is the centroid of cluster C_j (the mean vector of all the data points in C_j),
  - $d(\mathbf{x}, \mathbf{m}_j)$ is the (Eucledian) distance between data point $\mathbf{x}$ and centroid $\mathbf{m}_j$.

Seed choice

Results can vary drastically based on random seed selection

Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings

Common heuristics

- Random centers in the space
- Randomly pick examples
- Points least similar to any existing center (furthest centers heuristic)
- **Try out multiple starting points**
- Initialize with the results of another clustering method

K-means Clustering

- We now consider an example of iterative optimization algorithm for the special case of J_{SSE} objective function

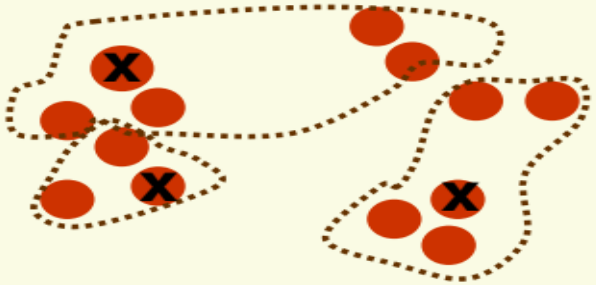
$$J_{SSE} = \sum_{i=1}^{k} \sum_{x \in D_i} || x - \mu_i ||^2$$

  - for a different objective function, we need a different optimization algorithm, of course
- Fix number of clusters to k ($c = k$)
- k-means is probably the most famous clustering algorithm
  - it has a smart way of moving from current partitioning to the next one

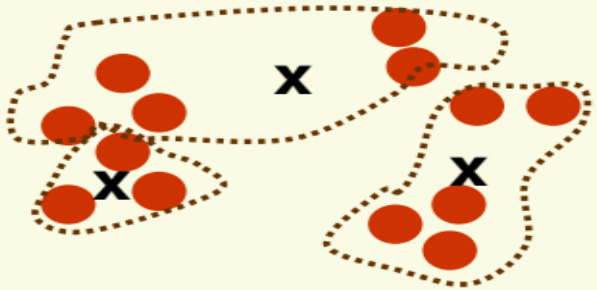
K-means Clustering

$k = 3$

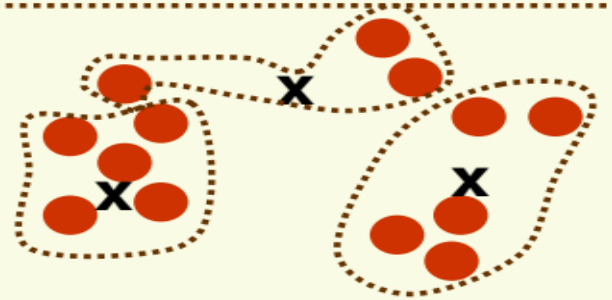
1. Initialize
   - pick ***k*** cluster centers arbitrary
   - assign each example to closest center



2. compute sample means for each cluster



3. reassign all samples to the closest mean

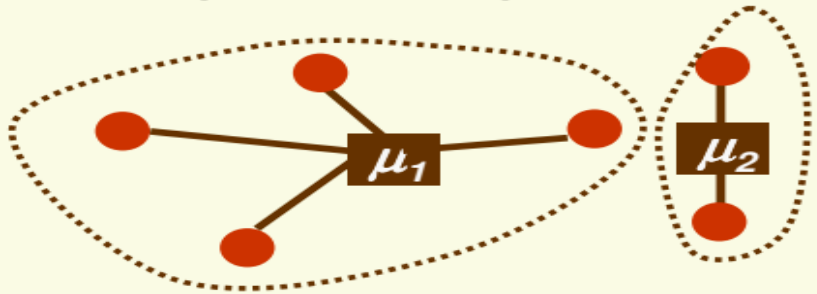


4. if clusters changed at step 3, go to step 2

K-means Clustering

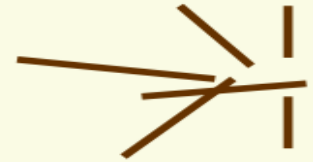
- Consider steps *2* and *3* of the algorithm

2. compute sample means for each cluster

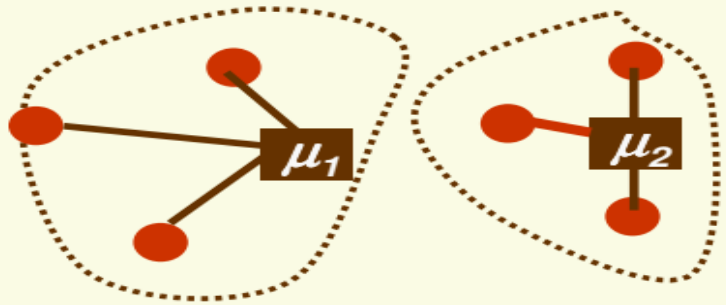


$$J_{SSE} = \sum_{i=1}^{k} \sum_{x \in D_i} || x - \mu_i ||^2$$

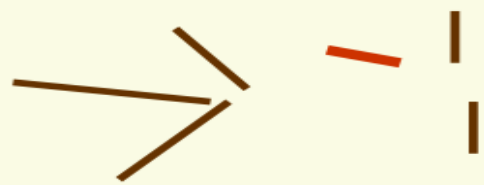
= sum of



3. reassign all samples to the closest mean

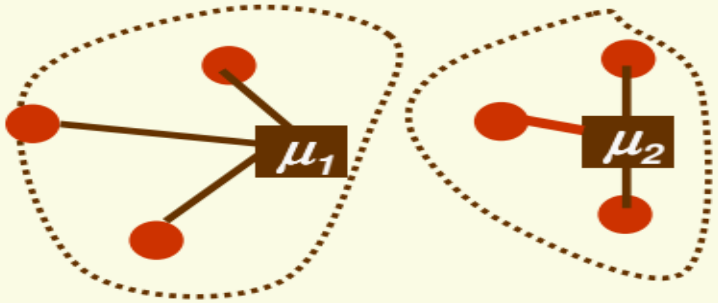


If we represent clusters by their old means, the error has gotten smaller

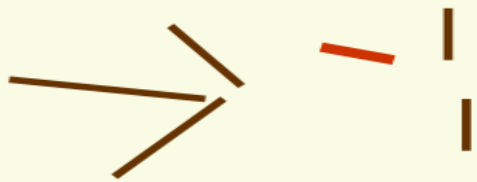


K-means Clustering

3. reassign all samples to the closest mean



If we represent clusters by their old means, the error has gotten smaller



- However we represent clusters by their new means, and mean is always the smallest representation of a cluster

$$\frac{\partial}{\partial z}\sum_{x\in D_i}\frac{1}{2}\| x-z\|^2=\frac{\partial}{\partial z}\sum_{x\in D_i}\frac{1}{2}\left(\| x\|^2-2x^tz+\| z\|^2\right)=\sum_{x\in D_i}(-x+z)=0$$

$$\Rightarrow z=\frac{1}{n_i}\sum_{x\in D_i}x$$

A Simple example showing the implementation of k-means algorithm
(using K=2)

Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

Step 1:

Initialization: Randomly we choose following two centroids (k=2) for two clusters.

In this case the 2 centroid are: m1=(1.0,1.0) and m2=(5.0,7.0).

Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

	Individual	Mean Vector
Group 1	1	(1.0, 1.0)
Group 2	4	(5.0, 7.0)

Step 2:

- Thus, we obtain two clusters containing:

  {1,2,3} and {4,5,6,7}.

- Their new centroids are:

Individual	Centroid 1	Centroid 2
1	0	7.21
2 (1.5, 2.0)	1.12	6.10
3	3.61	3.61
4	7.21	0
5	4.72	2.5
6	5.31	2.06
7	4.30	2.92

$$m_1 = \left(\frac{1}{3}(1.0+1.5+3.0), \frac{1}{3}(1.0+2.0+4.0)\right) = (1.83, 2.33)$$

$$m_2 = \left(\frac{1}{4}(5.0+3.5+4.5+3.5), \frac{1}{4}(7.0+5.0+5.0+4.5)\right)$$

$$= (4.12, 5.38)$$

$$d(m_1, 2) = \sqrt{|1.0-1.5|^2 + |1.0-2.0|^2} = 1.12$$

$$d(m_2, 2) = \sqrt{|5.0-1.5|^2 + |7.0-2.0|^2} = 6.10$$

Step 3:

- Now using these centroids we compute the Euclidean distance of each object, as shown in table.

Individual	Centroid 1	Centroid 2
1	1.57	5.38
2	0.47	4.28
3	2.04	1.78
4	5.64	1.84
5	3.15	0.73
6	3.78	0.54
7	2.74	1.08

- Therefore, the new clusters are:

  {1,2} and {**3**,4,5,6,7}

- Next centroids are: m1=(1.25,1.5) and m2 = (3.9,5.1)

- Step 4 :

  The clusters obtained are:

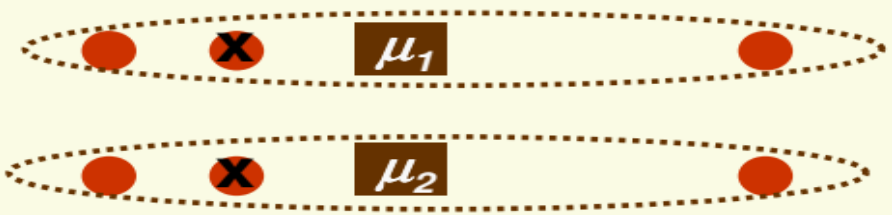
  {1,2} and {3,4,5,6,7}

- Therefore, there is no change in the cluster.
- Thus, the algorithm comes to a halt here and final result consist of 2 clusters {1,2} and {3,4,5,6,7}.

Individual	Centroid 1	Centroid 2
1	0.56	5.02
2	0. 56	3.92
3	3.05	1.42
4	6.66	2.20
5	4.16	0.41
6	4.78	0.61
7	3.75	0.72

K-means Clustering

- We just proved that by doing steps *2* and *3*, the objective function goes down
  - in two step, we found a “smart “ move which decreases the objective function
- Thus the algorithm converges after a finite number of iterations of steps *2* and *3*
- However the algorithm is not guaranteed to find a global minimum



2-means gets stuck here

global minimum of J_{SSE}

K-means Clustering

- Finding the optimum of J_{SSE} is NP-hard
- In practice, **k**-means clustering performs usually well
- It is very efficient
- Its solution can be used as a starting point for other clustering algorithms
- Still 100's of papers on variants and improvements of **k**-means clustering every year

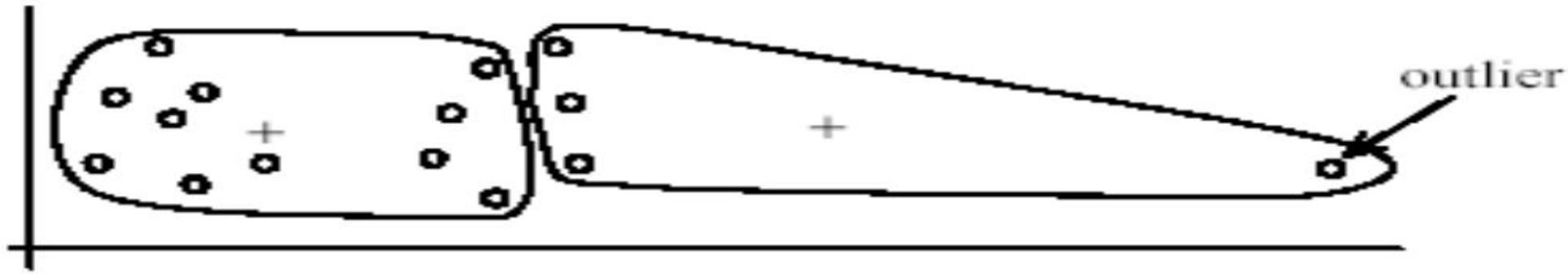
Why use K-means?

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time complexity: $O(tkn)$,
    where n is the number of data points,
    k is the number of clusters, and
    t is the number of iterations.
  - Since both k and t are small. k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.
- Note that: it terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity.

Weaknesses of K-means

- The algorithm is only applicable if the mean is defined.
  - For categorical data, *k*-mode - the centroid is represented by most frequent values.
- The user needs to specify *k*.
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.

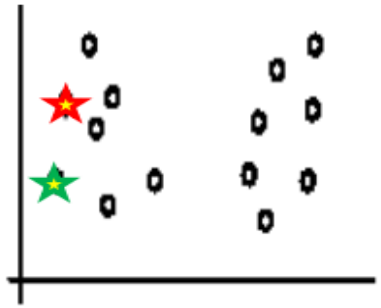
Outliers



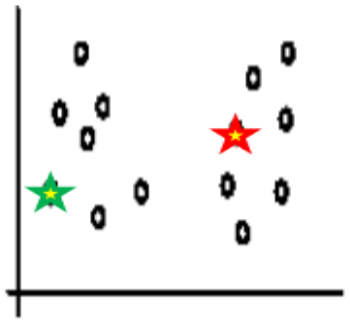
(A): Undesirable clusters

(B): Ideal clusters

Sensitivity to initial seeds

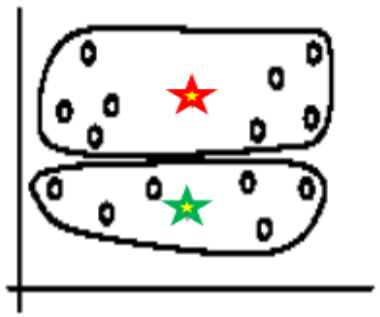


Random selection of seeds (centroids)

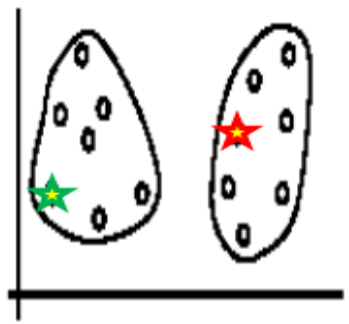


Random selection of seeds (centroids)

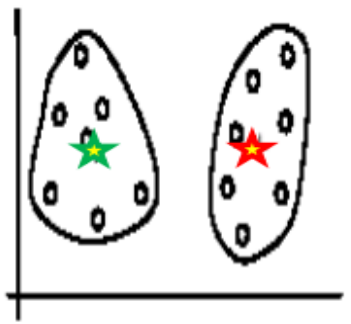
Iteration 1



Iteration 2



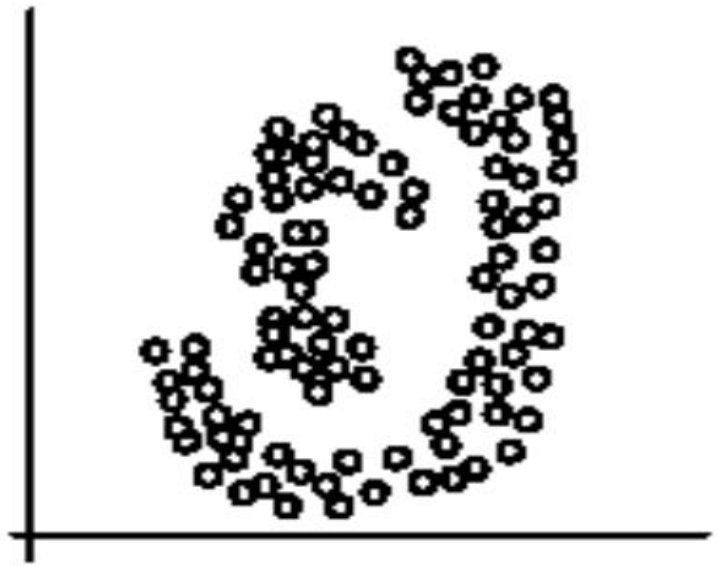
Iteration 1



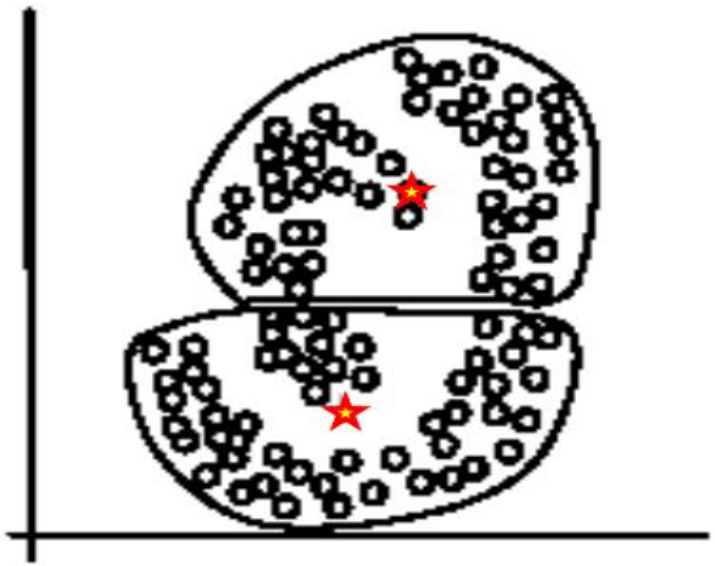
Iteration 2

Special data structures

- The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



(A): Two natural clusters



(B): *k*-means clusters