Clustering

CE-717: Machine Learning
Sharif University of Technology
Spring 2016

Soleymani

Outline

- Clustering Definition
- Clustering main approaches
  - Partitional (flat)
  - Hierarchical
- Clustering validation

Unsupervised learning

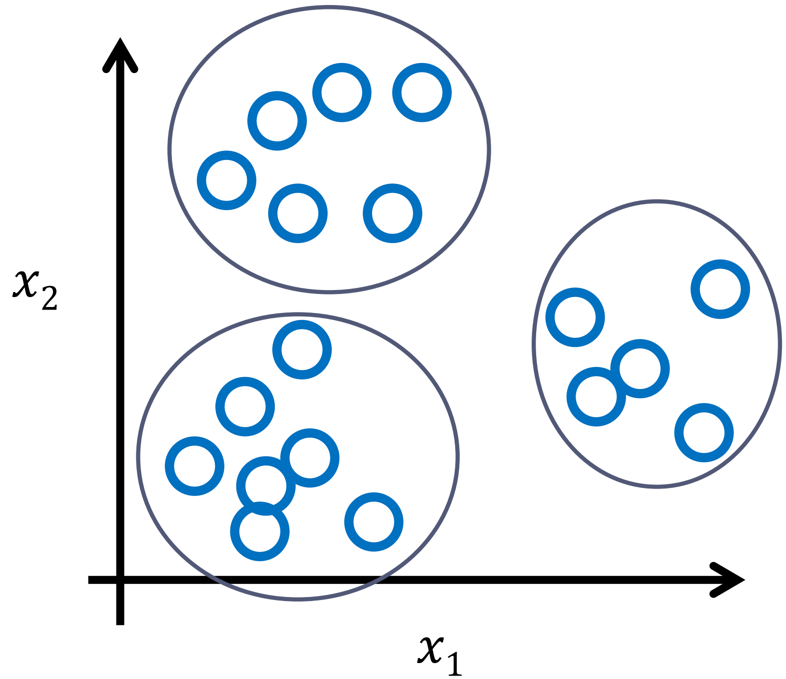
- **Clustering**: partitioning of data into groups of similar data points.

- **Density estimation**
  - Parametric & non-parametric density estimation

- **Dimensionality reduction**: data representation using a smaller number of dimensions while preserving (perhaps approximately) some properties of the data.

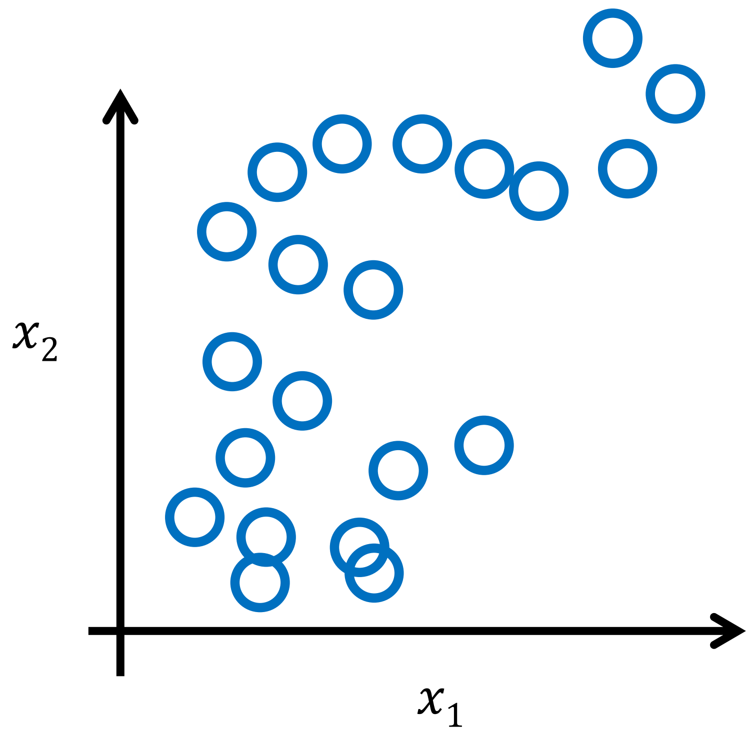
Clustering: Definition

- We have a set of unlabeled data points $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ and we intend to **find groups of similar objects** (based on the observed features)
  - high intra-cluster similarity
  - low inter-cluster similarity



Clustering: Another Definition

- Density-based definition:
  - Clusters are regions of high density that are separated from one another by regions of low density



Clustering Purpose

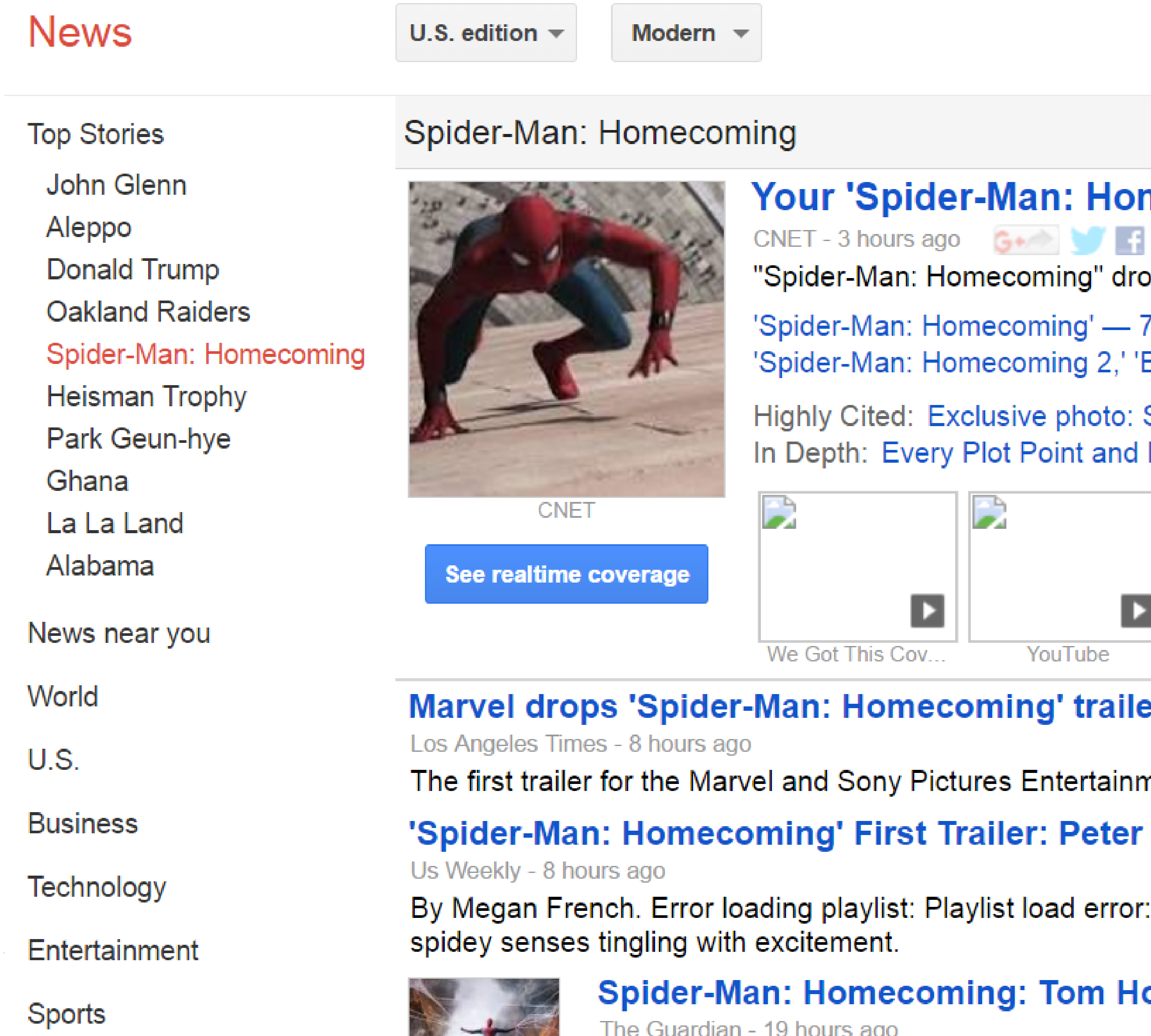
- **Preprocessing stage** to index, compress, or reduce the data
- Representing high-dimensional data in a low-dimensional space (e.g., for visualization purposes).
- As a tool to **understand the hidden structure** in data or to **group** them
  - To gain insight into the structure of the data prior to classifier design
  - To group the data when no label is available

Clustering Applications

- Information retrieval (search and browsing)
  - Cluster text docs or images based on their content
  - Cluster groups of users based on their access patterns on webpages

Clustering of docs

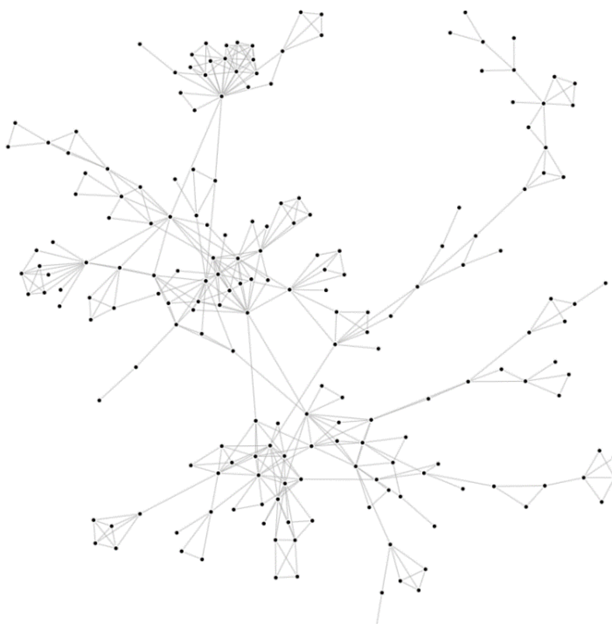
- Google news

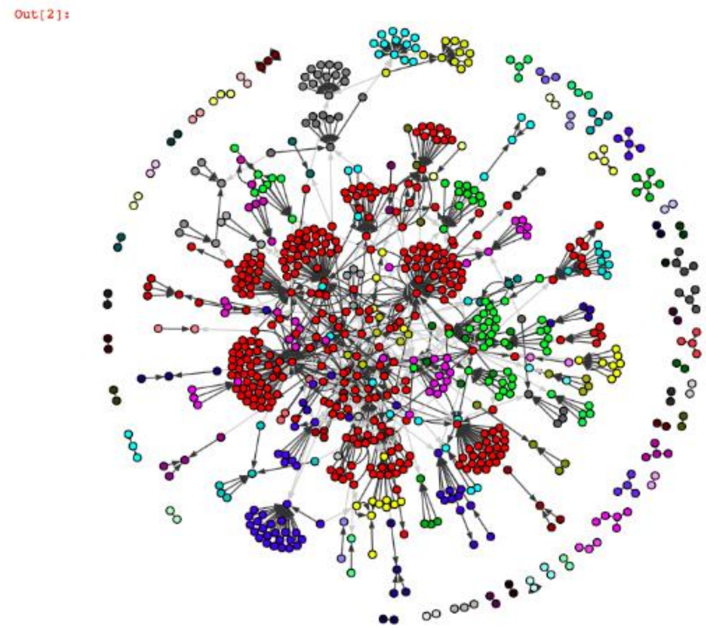


Clustering Applications

- Information retrieval (search and browsing)
  - Cluster text docs or images based on their content
  - Cluster groups of users based on their access patterns on webpages
- **Cluster users of social networks** by interest (community detection).

Social Network: Community Detection



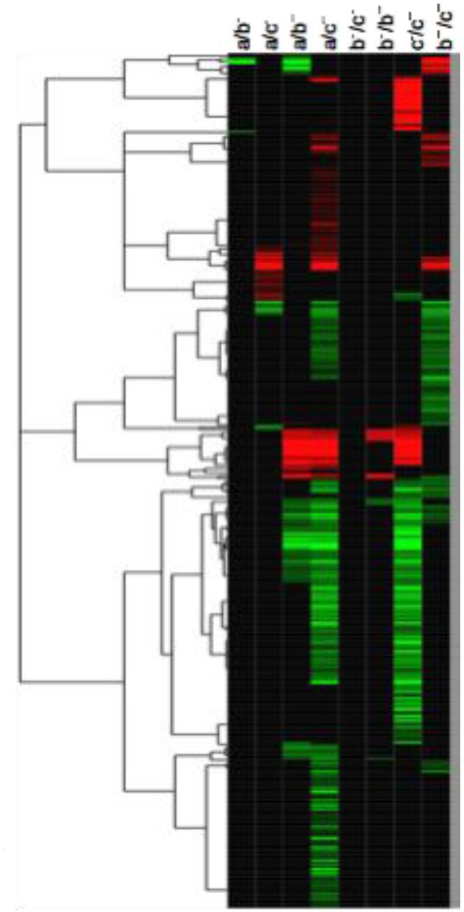


Clustering Applications

- Information retrieval (search and browsing)
  - Cluster text docs or images based on their content
  - Cluster groups of users based on their access patterns on webpages
- Cluster users of social networks by interest (community detection).
- **Bioinformatics**
  - cluster similar proteins together (similarity wrt chemical structure and/or functionality etc)
  - or cluster similar genes according to microarray data

Gene clustering

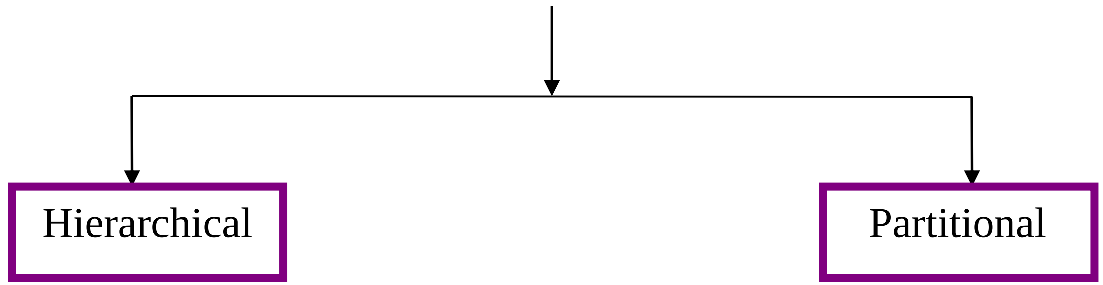
- Microarrays measures the expression of all genes
- Clustering genes can help determine new functions for unknown genes



Clustering Applications

- Information retrieval (search and browsing)
  - Cluster text docs or images based on their content
  - Cluster groups of users based on their access patterns on webpages
- Cluster users of social networks by interest (community detection).
- Bioinformatics
  - Cluster similar proteins together (similarity wrt chemical structure and/or functionality etc) or similar genes according to microarray data
- **Market segmentation**
  - Clustering customers based on the their purchase history and their characteristics
- Image segmentation
- Many more applications

Categorization of Clustering Algorithms



Partitional algorithms: Construct various partitions and then evaluate them by some criterion

Hierarchical algorithms: Create a hierarchical decomposition of the set of objects using some criterion

Clustering methods we will discuss

- Objective based clustering
  - K-means
  - EM-style algorithm for clustering for mixture of Gaussians (in the next lecture)

- Hierarchical clustering

Partitional Clustering

- $\mathcal{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$
- $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_K\}$
  - $\forall j, \mathcal{C}_j \neq \emptyset$
  - $\bigcup_{j=1}^{K} \mathcal{C}_j = \mathcal{X}$
  - $\forall i, j,\ \mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ (disjoint partitioning for hard clustering)

Nonhierarchical, each instance is placed in exactly one of K non-overlapping clusters.

Hard clustering: Each data can belong to one cluster only

- Since the output is only one set of clusters the user has to specify the desired number of clusters K.

Partitioning Algorithms: Basic Concept

- Construct a partition of a set of N objects into a set of K clusters
  - The number of clusters K is given in advance
  - Each object belongs to **exactly one** cluster in hard clustering methods

- K-means is the most popular partitioning algorithm

Objective Based Clustering

- **Input**: A set of N points, also a distance/dissimilarity measure
- **Output**: a partition of the data.

- **k-median**: find center pts $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K$ to minimize

$$\sum_{i=1}^{N} \min_{j \in 1,\dots,K} d(\boldsymbol{x}^{(i)}, \boldsymbol{c}_j)$$

- **k-means**: find center pts $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K$ to minimize

$$\sum_{i=1}^{N} \min_{j \in 1,\dots,K} d^2(\boldsymbol{x}^{(i)}, \boldsymbol{c}_j)$$

- **k-center**: find partition to minimize the maxim radius

Distance Measure

- Let O_1 and O_2 be two objects from the universe of possible objects. The distance (dissimilarity) between O_1 and O_2 is a real number denoted by $d(O_1, O_2)$

- Specifying the distance $d(\boldsymbol{x}, \boldsymbol{x}')$ between pairs $(\boldsymbol{x}, \boldsymbol{x}')$.
  - E.g., # keywords in common, edit distance
  - Example: Euclidean distance in the space of features

K-means Clustering

- **Input**: a set $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(N)}$ of data points (in a d-dim feature space) and an integer K

- **Output**: a set of K representatives $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K \in \mathbb{R}^d$ as the cluster representatives
  - data points are assigned to the clusters according to their distances to $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K$
    - Each data is assigned to the cluster whose representative is nearest to it

- **Objective**: choose $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K$ to minimize:

$$\sum_{i=1}^{N} \min_{j \in 1,\dots,K} d^2(\boldsymbol{x}^{(i)}, \boldsymbol{c}_j)$$

Euclidean k-means Clustering

- **Input**: a set $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(N)}$ of data points (in a d-dim feature space) and an integer K

- **Output**: a set of K representatives $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K \in \mathbb{R}^d$ as the cluster representatives
  - data points are assigned to the clusters according to their distances to $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K$
    - Each data is assigned to the cluster whose representative is nearest to it

- **Objective**: choose $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K$ to minimize:

$$\sum_{i=1}^{N} \min_{j \in 1,\dots,K} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2$$

each point assigned to its closest cluster representative

Euclidean k-means Clustering: Computational Complexity

- To find the optimal partition, we need to exhaustively enumerate all partitions
  - In how many ways can we assign k labels to N observations?

- NP hard: even for $k = 2$ or $d = 2$

- For k=1: $\min_{\boldsymbol{c}} \sum_{i=1}^{N} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c} \right\|^2$
  - $\boldsymbol{c} = \boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}^{(i)}$

- For $d = 1$, dynamic programming in time $O(N^2 K)$.

Common Heuristic in Practice: The Lloyd's method

- Input: A set $\mathcal{X}$ of N datapoints $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(N)}$ in $\mathbb{R}^d$

- **Initialize** centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K \in \mathbb{R}^d$ in any way.
- **Repeat** until there is no further change in the cost.
  - For each j: $\mathcal{C}_j \leftarrow \{\boldsymbol{x} \in \mathcal{X} | \text{where } \boldsymbol{c}_j \text{ is the closest center to } \boldsymbol{x}\}$
  - For each j: $\boldsymbol{c}_1 \leftarrow$mean of members of $\mathcal{C}_j$

Holding centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K$ fixed
Find optimal assignments $\mathcal{C}_1, \dots, \mathcal{C}_K$ of data points to clusters

Holding cluster assignments $\mathcal{C}_1, \dots, \mathcal{C}_K$ fixed
Find optimal centers $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_K$

K-means Algorithm (The Lloyd's method)

Select k random points $\boldsymbol{c}_1, \boldsymbol{c}_2, \dots \boldsymbol{c}_k$ as clusters' initial centroids.

Repeat until *converges* (or other stopping criterion):

for i=1 to N do:

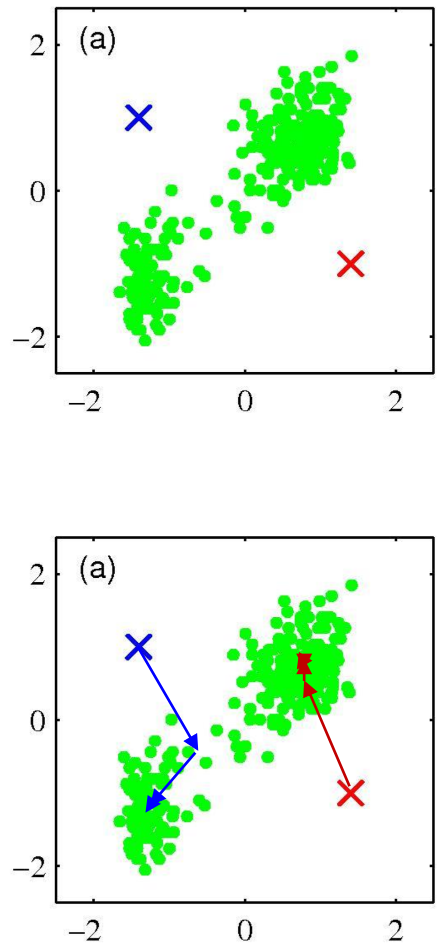
Assign $\boldsymbol{x}^{(i)}$ to the closet cluster and thus $\mathcal{C}_j$ contains all data that are closer to $\boldsymbol{c}_j$ than to anyother cluster

for j=1 to k do

$$\boldsymbol{c}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \boldsymbol{x}^{(i)}$$

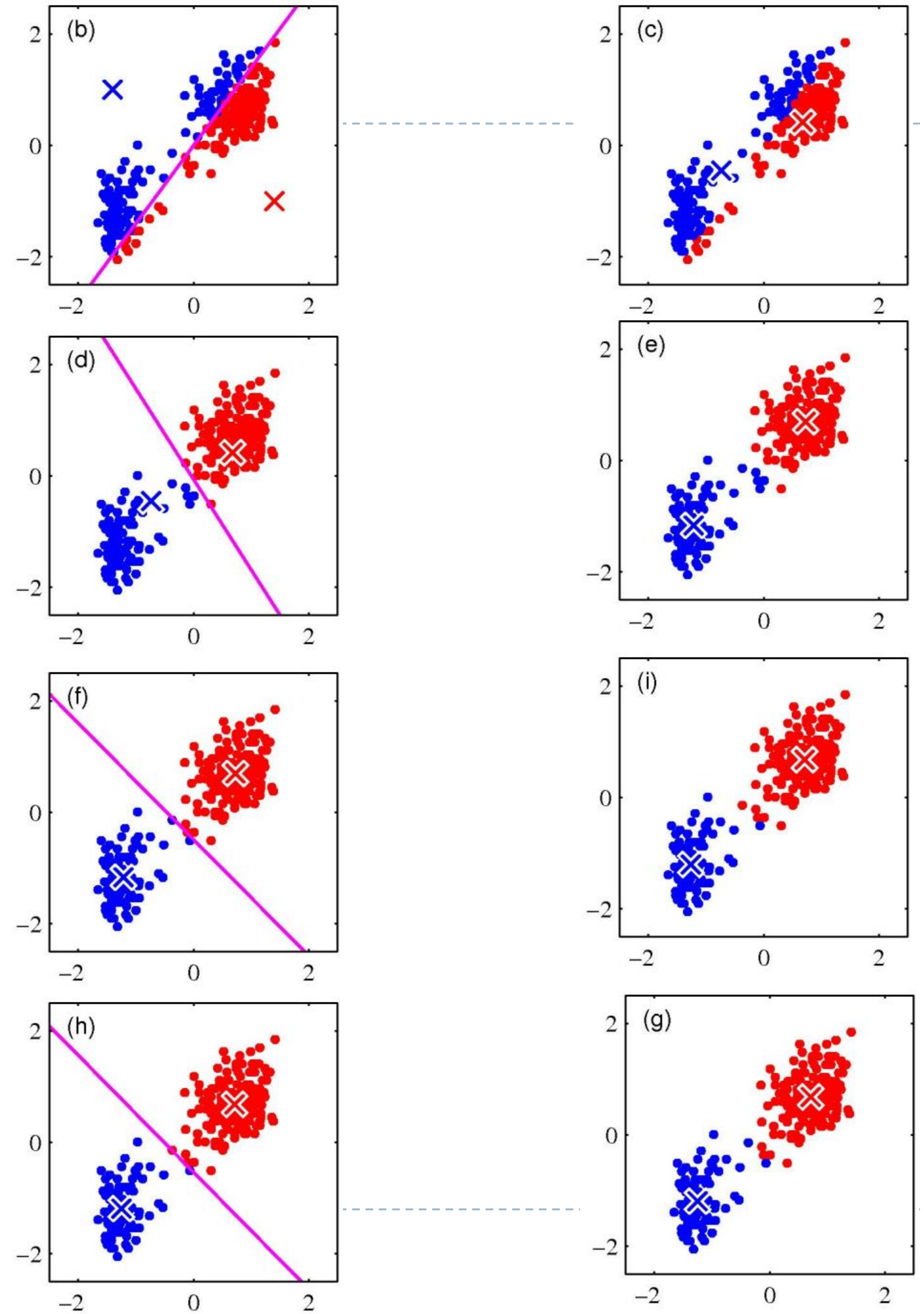
Assign data based on current centers

Re-estimate centers based on current assignment



Assigning data to clusters

Updating means



[Bishop]

Intra-cluster similarity

- k-means optimizes intra-cluster similarity:

$$J(\mathcal{C}) = \sum_{j=1}^{K} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2$$

$$\boldsymbol{c}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \boldsymbol{x}^{(i)}$$

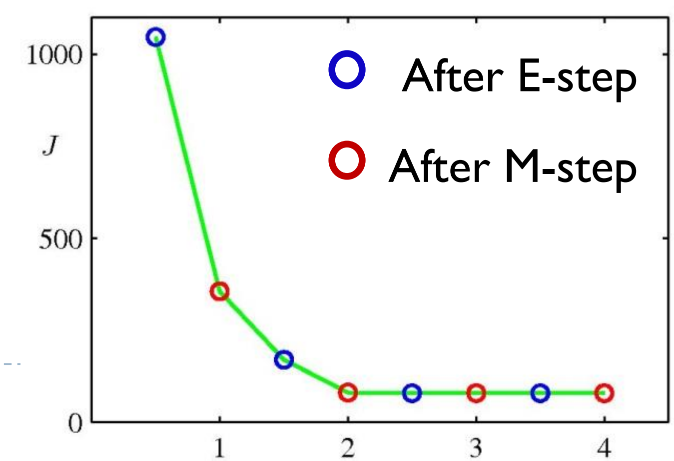
$$\sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2 = \frac{1}{2|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \sum_{\boldsymbol{x}^{(i')} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{x}^{(i')} \right\|^2$$

the average distance to members of the same cluster

K-means: Convergence

- It always converges.

- Why should the *K*-means algorithm ever reach a state in which clustering doesn't change.
  - Reassignment stage monotonically decreases J since each vector is assigned to the closest centroid.
  - Centroid update stage also for each cluster minimizes the sum of squared distances of the assigned points to the cluster from its center.



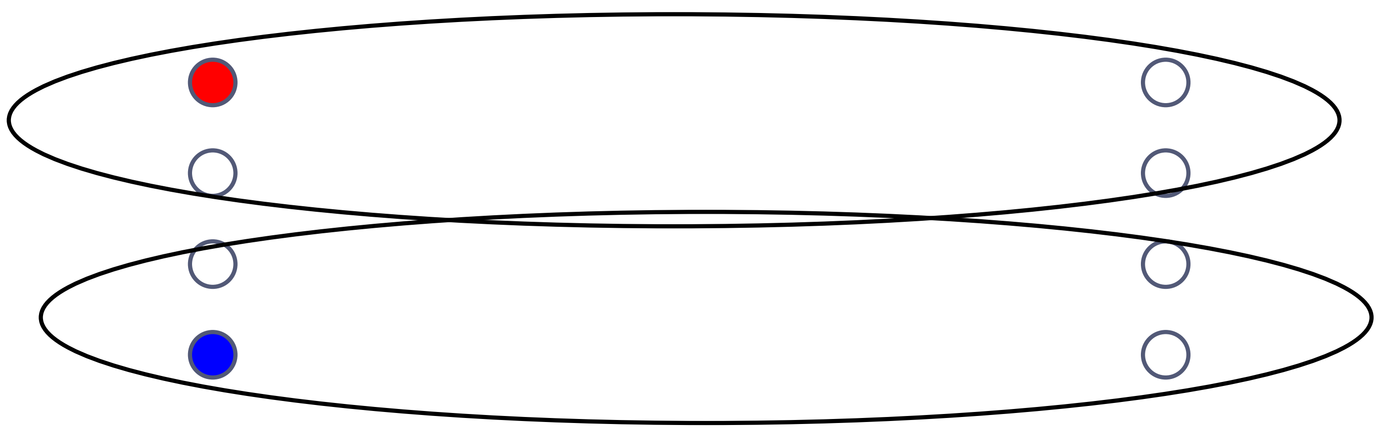
[Bishop]

Local optimum

- It always converges
- but it may converge at a local optimum that is different from the global optimum
  - may be arbitrarily worse in terms of the objective score.

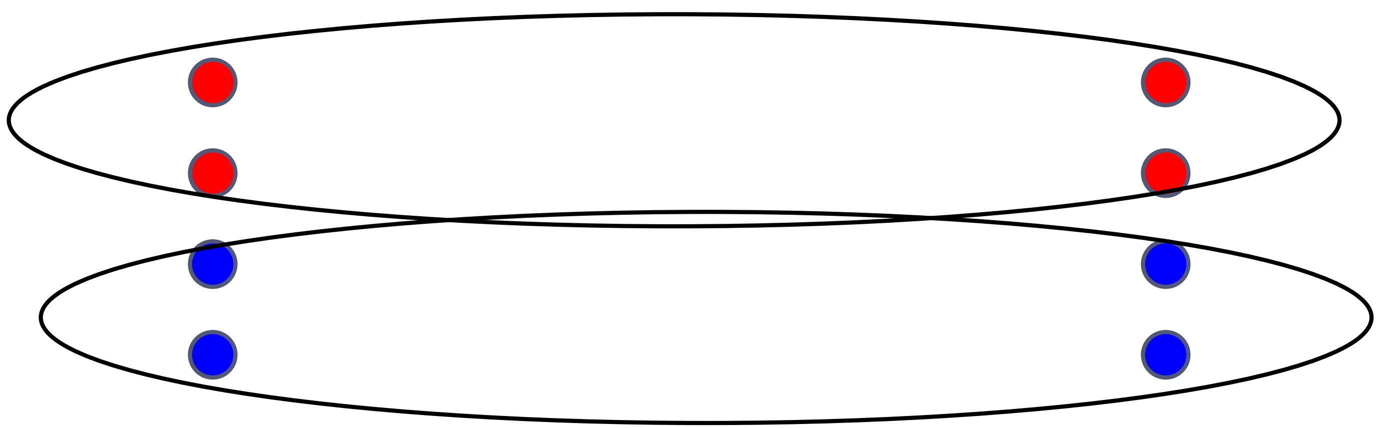
Local optimum

- It always converges
- but it may converge at a local optimum that is different from the global optimum
  - may be arbitrarily worse in terms of the objective score.



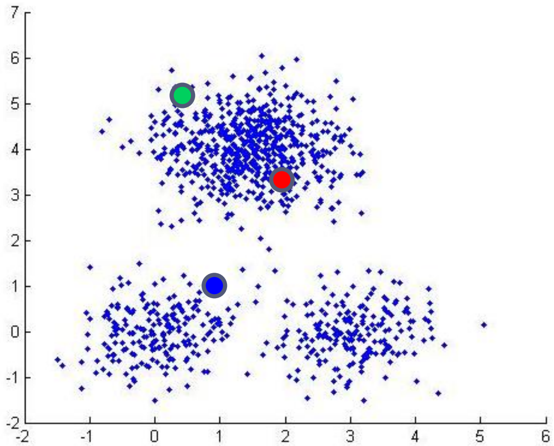
Local optimum

- It always converges
- but it may converge at a local optimum that is different from the global optimum
  - may be arbitrarily worse in terms of the objective score.

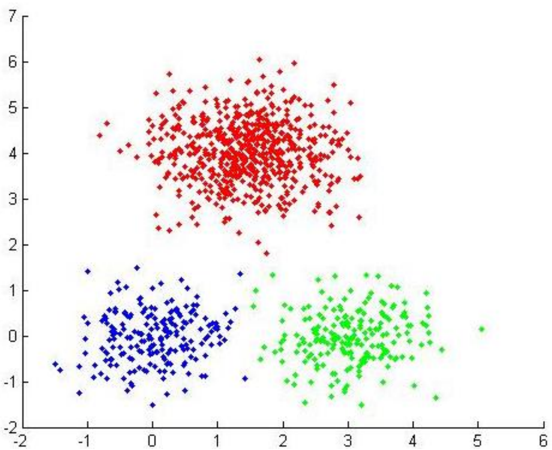


Local optimum: every point is assigned to its nearest center and every center is the mean value of its points.

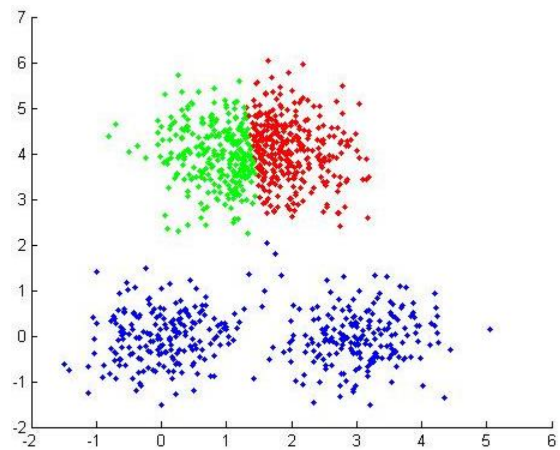
K-means: Local Minimum Problem



Original Data



Optimal Clustering



The obtained Clustering

The Lloyd's method: Initialization

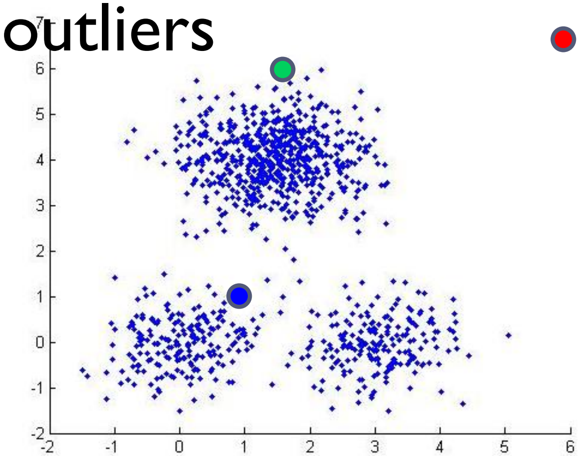
- Initialization is crucial (how fast it converges, quality of clustering)
  - Random centers from the data points
    - Multiple runs and select the best ones
  - Initialize with the results of another method
  - Select good initial centers using a heuristic
    - Furthest traversal
    - K-means ++ (works well and has provable guarantees)

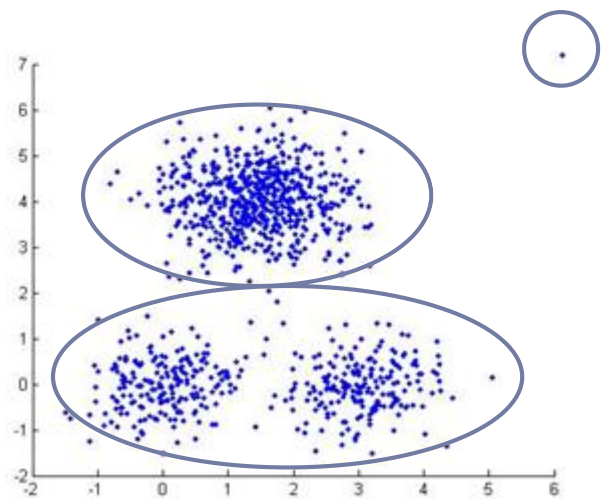
Another Initialization Idea: Furthest Point Heuristic

- Choose $\boldsymbol{c}_1$ arbitrarily (or at random).
- For $j = 2, \dots, K$
  - Select $\boldsymbol{c}_j$ among datapoints $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(N)}$ that is farthest from previously chosen $\boldsymbol{c}_1, \dots, \boldsymbol{c}_{j-1}$

Another Initialization Idea: Furthest Point Heuristic

- It is sensitive to outliers





K-means++ Initialization: D2 sampling [AV07]

- Combine random initialization and furthest point initialization ideas
- Let the probability of selection of the point be proportional to the distance between this point and its nearest center.
  - probability of selecting of $\boldsymbol{x}$ is proportional to $D^2(\boldsymbol{x}) = \min_{k<j} \|\boldsymbol{x} - \boldsymbol{c}_k\|^2$.

- Choose $\boldsymbol{c}_1$ arbitrarily (or at random).
- For $j = 2, \dots, K$
  - Select $\boldsymbol{c}_j$ among data points $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(N)}$ according to the distribution:

$$\Pr(\boldsymbol{c}_j = \boldsymbol{x}^{(i)}) \propto \min_{k<j} \left\|\boldsymbol{x}^{(i)} - \boldsymbol{c}_k\right\|^2$$

- **Theorem:** K-means++ always attains an $O(\log k)$ approximation to optimal k-means solution in expectation.

How Many Clusters?

- Number of clusters k is given in advance in the k-means algorithm
  - However, finding the "right" number of clusters is a part of the problem

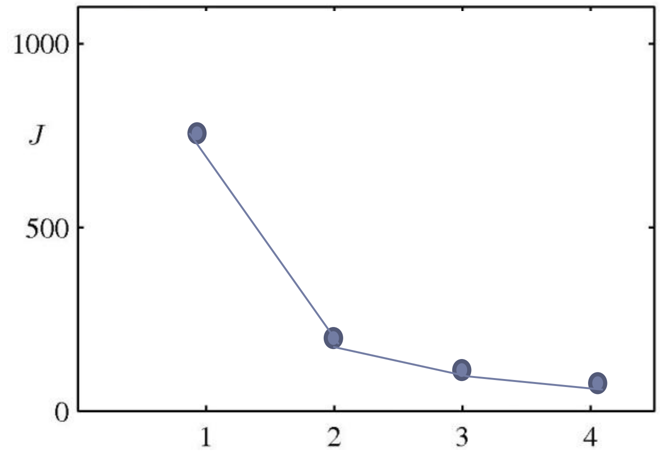
- Tradeoff between having better focus within each cluster and having too many clusters

- Hold-out validation/cross-validation on auxiliary task (e.g., supervised learning task).

- Optimization problem: penalize having lots of clusters
  - some criteria can be used to automatically estimate *k*
    - Penalize the number of bits you need to describe the extra parameter

$$J'(\mathcal{C}) = J(\mathcal{C}) + |\mathcal{C}| \times \log N$$

How Many Clusters?



- Heuristic: Find large gap between $k-1$-means cost and k-means cost.
  - "knee finding" or "elbow finding".

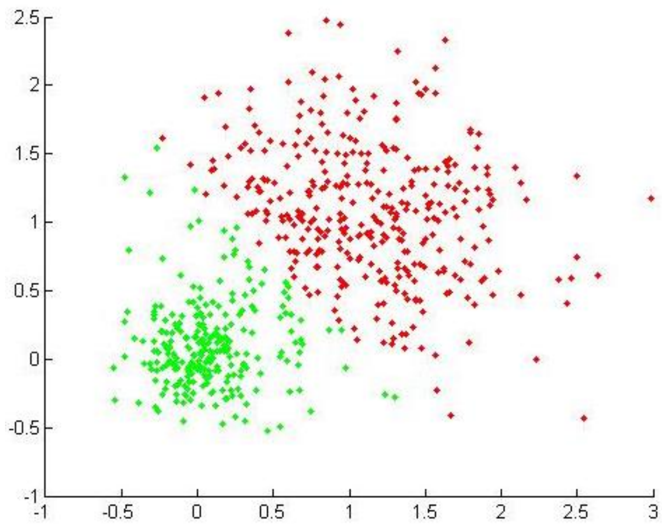
K-means: Advantages and disadvantages

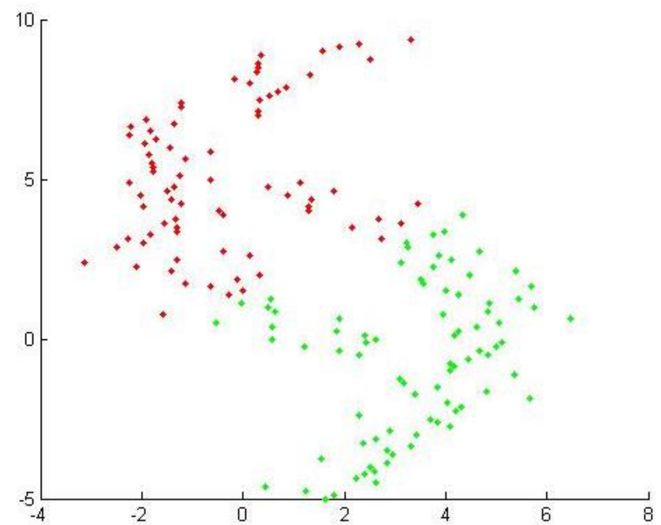
- Strength
  - It is a simple method
  - Relatively efficient: $O(tKNd)$, where t is the number of iterations.
    - Usually $t \ll n$.
    - *K*-means typically converges quickly

- Weakness
  - Need to specify *K*, the *number* of clusters, in advance
  - Often terminates at a *local optimum*.
  - Not suitable to discover clusters with arbitrary shapes
  - Works for numerical data. What about categorical data?
  - Noise and outliers can be considerable trouble to *K*-means

k-means Algorithm: Limitation

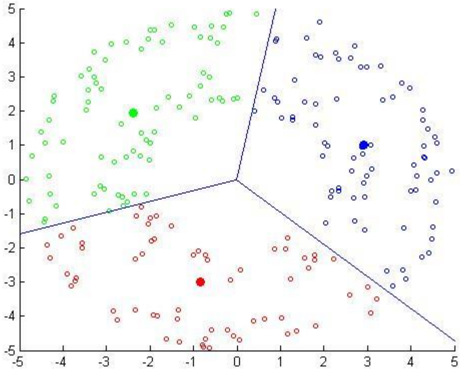
- In general, k-means is unable to find clusters of arbitrary shapes, sizes, and densities
  - Except to very distant clusters



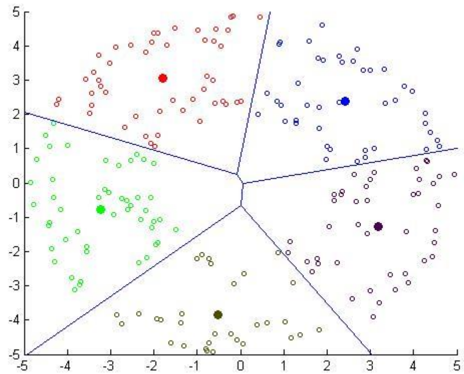


K-means: Vector Quantization

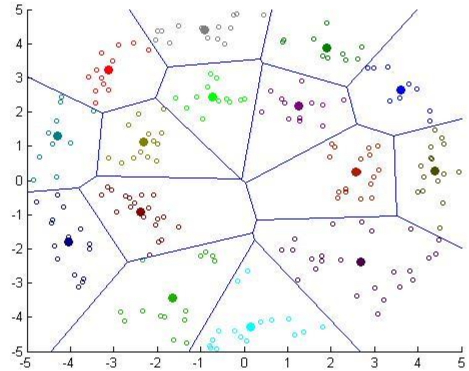
- Data Compression
  - Vector quantization: construct a codebook using k-means
    - cluster means as prototypes representing examples assigned to clusters.



$k = 3$



$k = 5$



$k = 15$

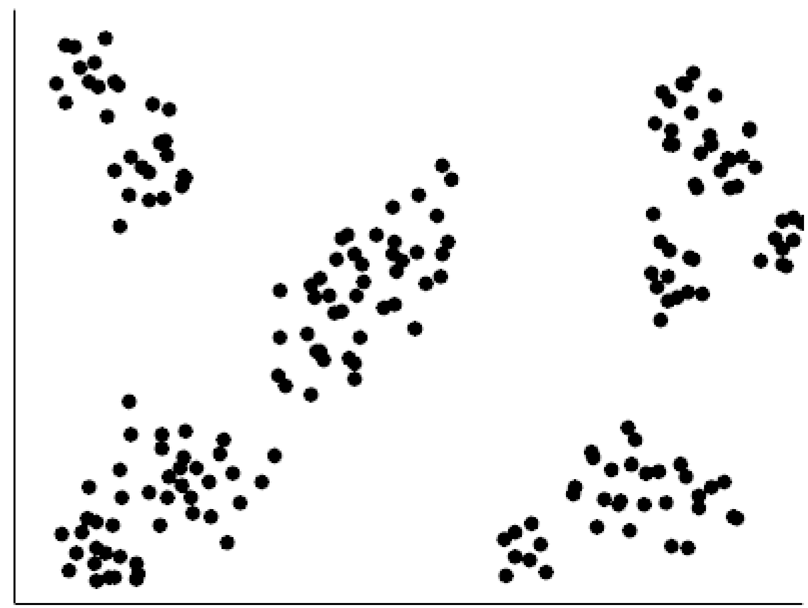
K-means

- K-means was proposed near 60 years ago
  - thousands of clustering algorithms have been published since then
  - However, K-means is still widely used.
- This speaks to the difficulty in designing a general purpose clustering algorithm and the ill-posed problem of clustering.

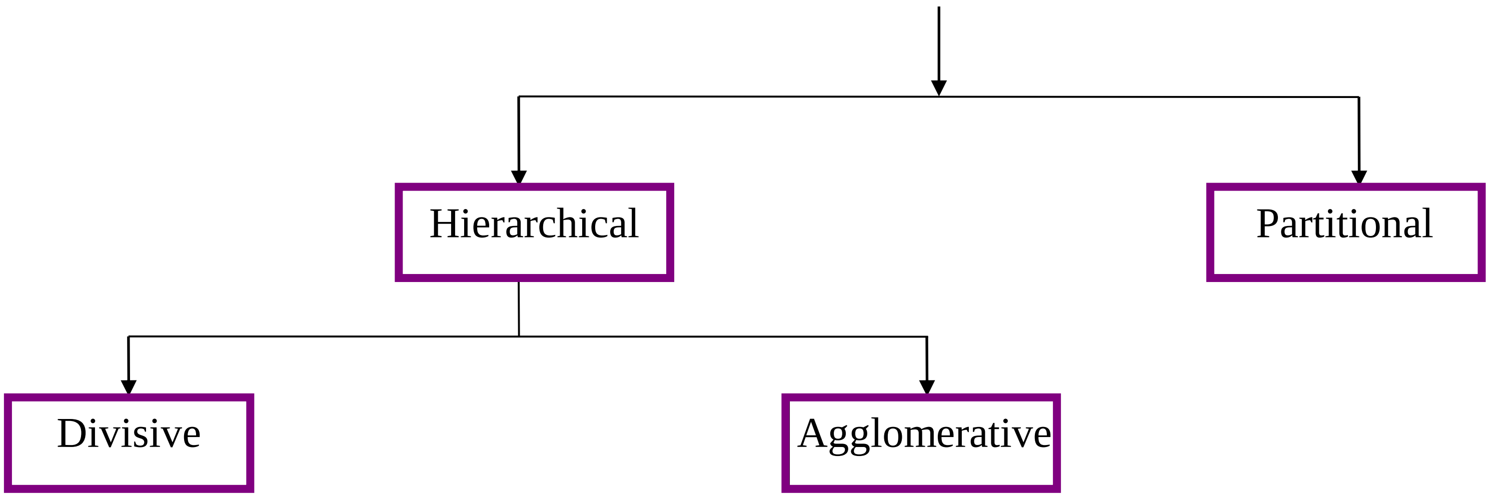
A.K. Jian, Data Clustering: 50 years beyond k-means,2010.

Hierarchical Clustering

- Notion of a cluster can be ambiguous?
- How many clusters?
- Hierarchical Clustering: Clusters contain sub-clusters and sub-clusters themselves can have sub-sub-clusters, and so on
  - Several levels of details in clustering
- A hierarchy might be more natural.
  - Different levels of granularity



Categorization of Clustering Algorithms



Hierarchical Clustering

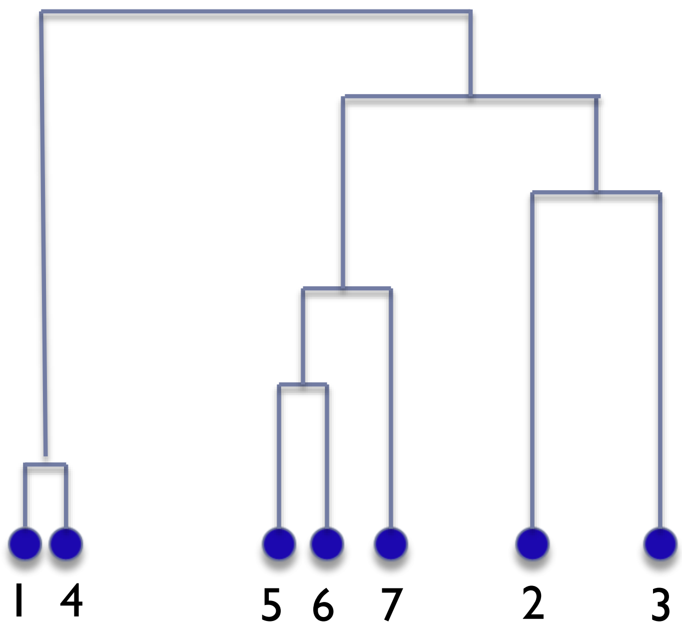
- Agglomerative (bottom up):
  - Starts with each data in a separate cluster
  - Repeatedly joins the closest pair of clusters, until there is only one cluster (or other stopping criteria).

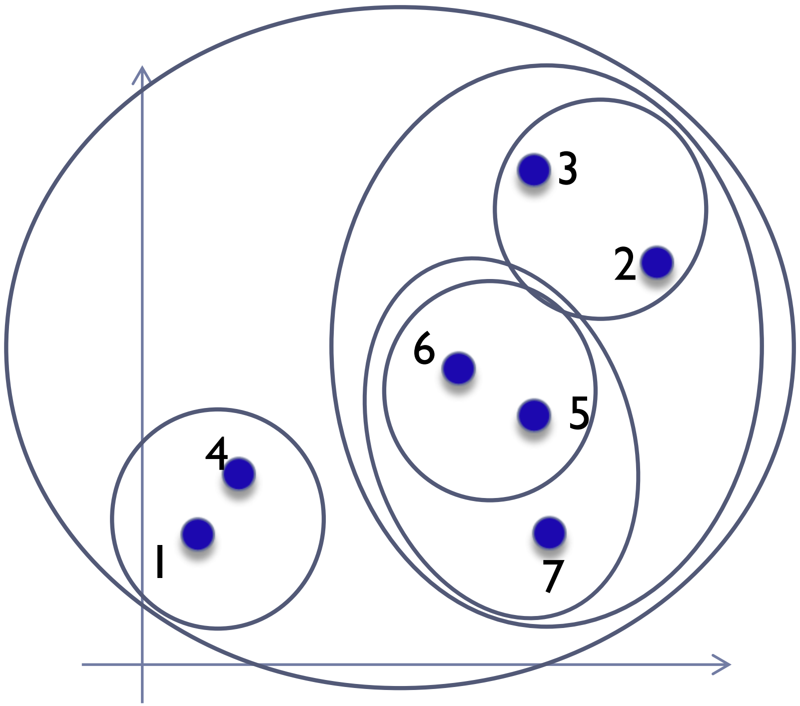
- Divisive (top down):
  - Starts with the whole data as a cluster
  - Repeatedly divide data in one of the clusters until there is only one data in each cluster (or other stopping criteria).

Example

- Hierarchical Agglomerative Clustering (HAC)

Height represents the distance at which the merge occurs

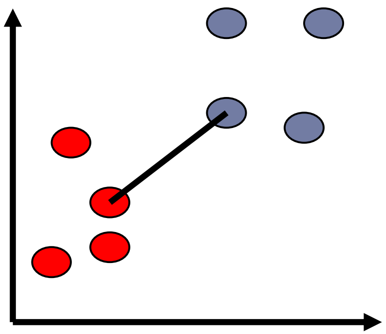




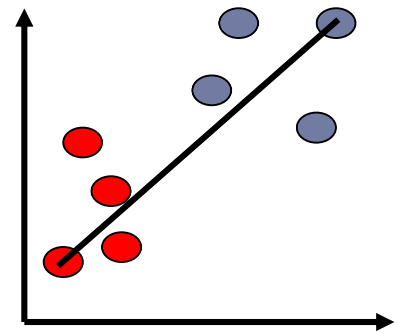
Distances between Cluster Pairs

- Many variants to defining distances between pair of clusters
  - **Single-link**
    - Minimum distance between different pairs of data
  - **Complete-link**
    - Maximum distance between different pairs of data
  - **Centroid**
    - Distance between centroids (centers of gravity)
  - **Average-link**
    - Average distance between pairs of elements

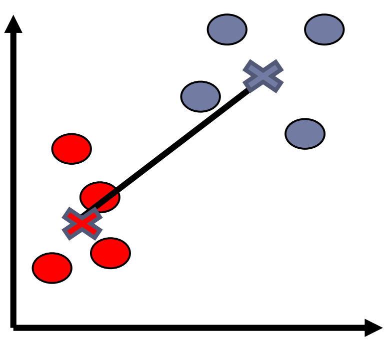
Distances between Cluster Pairs



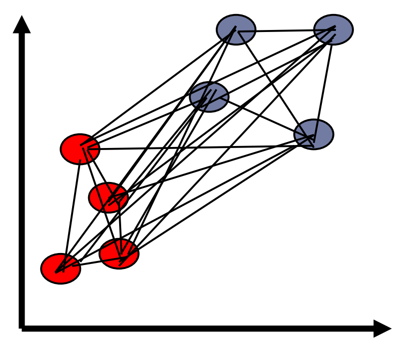
Single-link



Complete-link



Ward's



Average-link

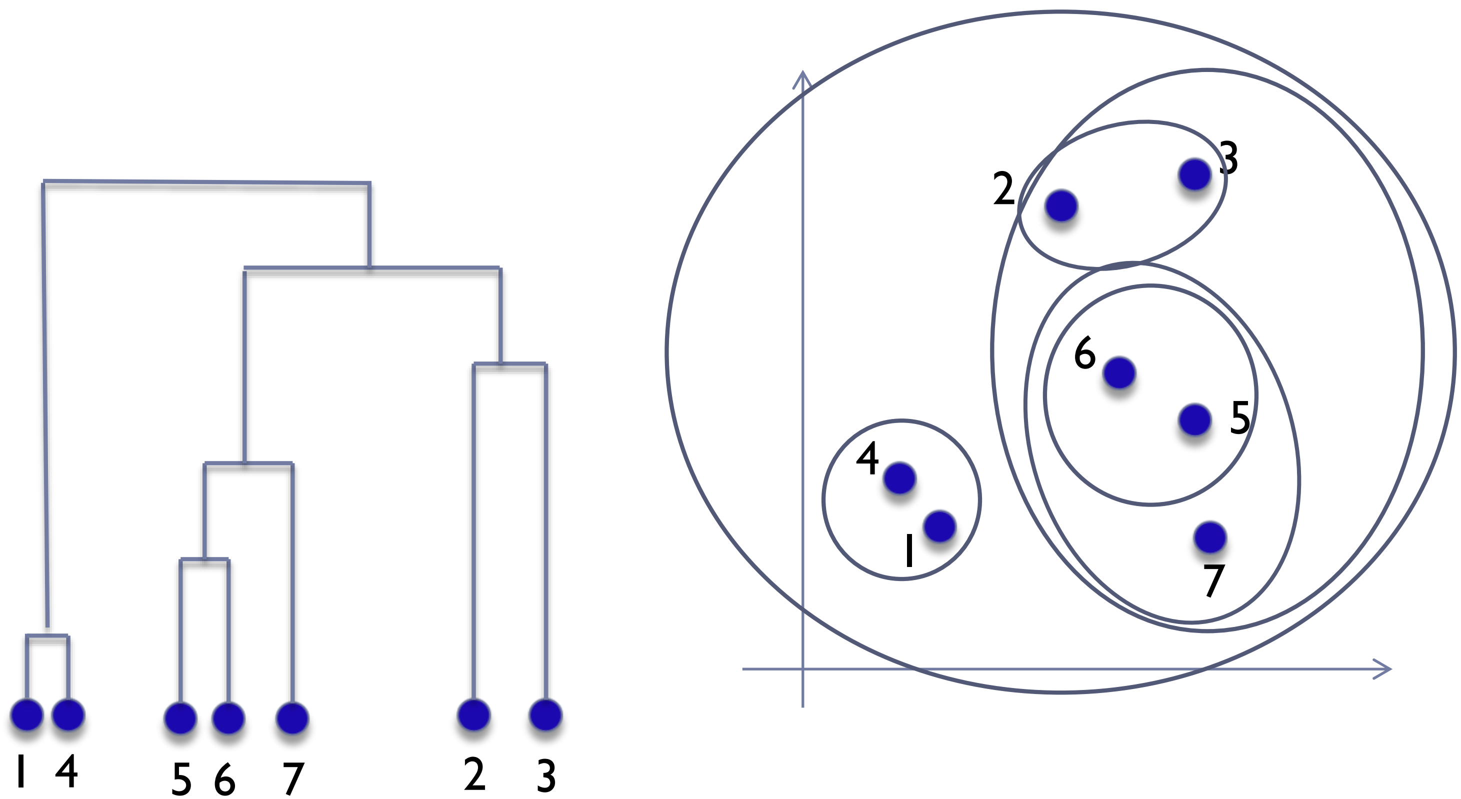
Single Linkage

- The minimum of all pairwise distances between points in the two clusters:

$$dist_{SL}(\mathcal{C}_i, \mathcal{C}_j) = \min_{\boldsymbol{x}\in\mathcal{C}_i,\, \boldsymbol{x}'\in\mathcal{C}_j} dist(\boldsymbol{x}, \boldsymbol{x}')$$

- "straggly" (long and thin) clusters due to chaining effect.

Single-Link



keep max bridge length as small as possible.

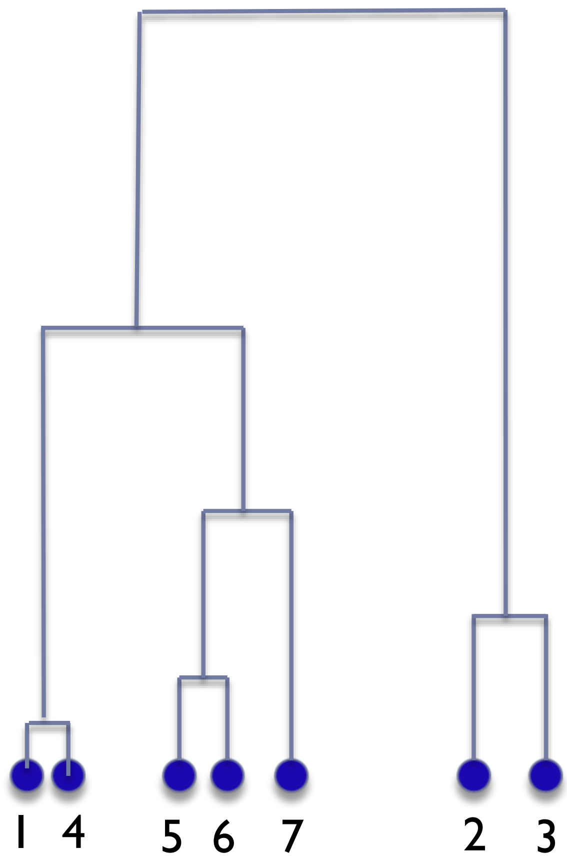
Complete Linkage

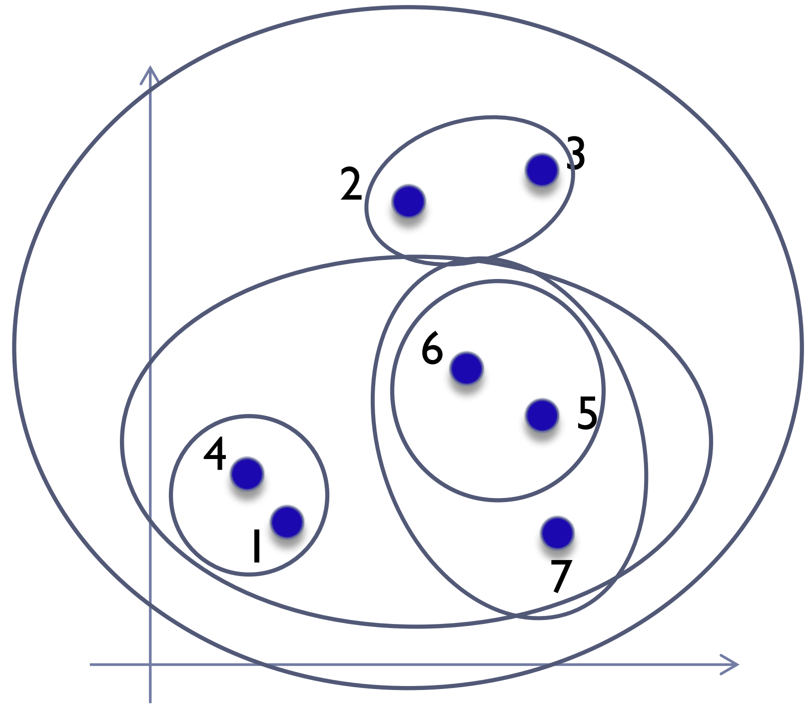
- The maximum of all pairwise distances between points in the two clusters:

$$dist_{CL}\left(\mathcal{C}_i, \mathcal{C}_j\right) = \max_{\boldsymbol{x}\in\mathcal{C}_i,\, \boldsymbol{x}'\in\mathcal{C}_j} dist(\boldsymbol{x}, \boldsymbol{x}')$$

- Makes "tighter," spherical clusters typically preferable.

Complete Link





keep max diameter as small as possible.

Ward's method

- The distances between centers of the two clusters (weighted to consider sizes of clusters too):

$$dist_{Ward}(\mathcal{C}_i, \mathcal{C}_j) = \frac{|\mathcal{C}_i||\mathcal{C}_j|}{|\mathcal{C}_i| + |\mathcal{C}_j|} dist(\boldsymbol{c}_i, \boldsymbol{c}_j)$$

- Merge the two clusters such that the increase in k-means cost is as small as possible.
- Works well in practice.

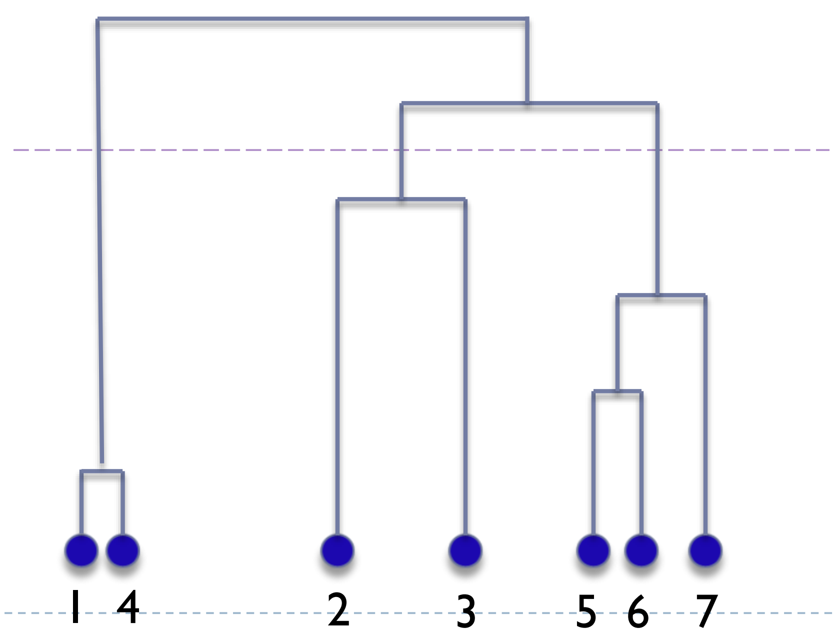
Computational Complexity

- In the first iteration, all HAC methods compute similarity of all pairs of N individual instances which is $O(N^2)$ similarity computation.
- In each $N-2$ merging iterations, compute the distance between the most recently created cluster and all other existing clusters.
- if done naively $O(N^3)$ but if done more cleverly $O(N^2 \log N)$

Dendrogram: Hierarchical Clustering

- Clustering obtained by cutting the dendrogram at a desired level
  - Cut at a pre-specified level of similarity
  - where the gap between two successive combination similarities is largest
  - select the cutting point that produces *K* clusters

Where to "cut" the dendrogram is user-determined.



K-means vs. Hierarchical

- Time cost:
  - K-means is usually fast while hierarchical methods do not scale well
- Human intuition
  - Hierarchical structure maps nicely onto human intuition for some domains and provides more natural output
- Local minimum problem
  - It is very common for k-means
  - However, hierarchical methods like any heuristic search algorithms also suffer from local optima problem.
    - Since they can never undo what was done previously
- Choosing of the number of clusters
  - There is no need to specify the number of clusters in advance for hierarchical methods