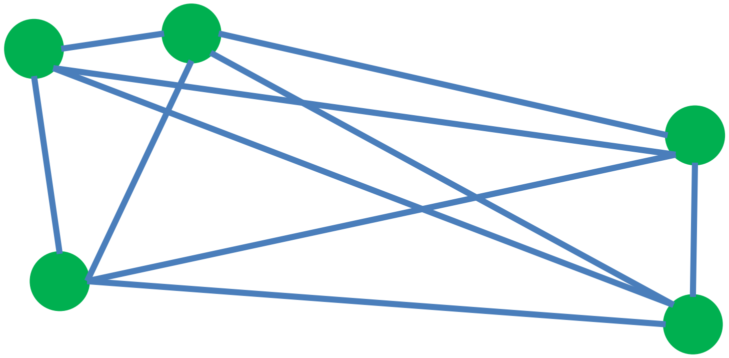
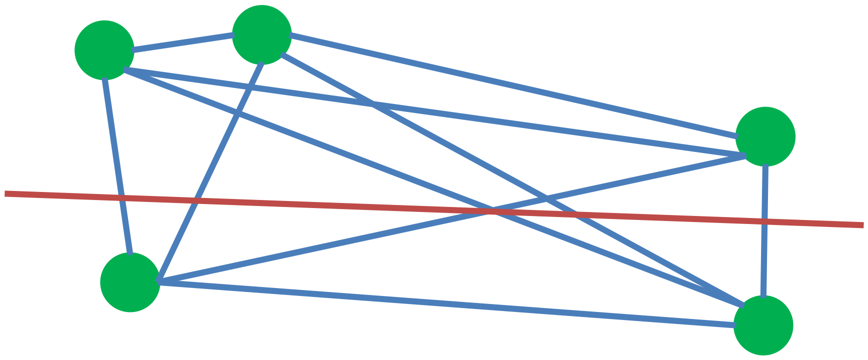
Clustering as graph cut

- Describe the pairwise distance via a graph



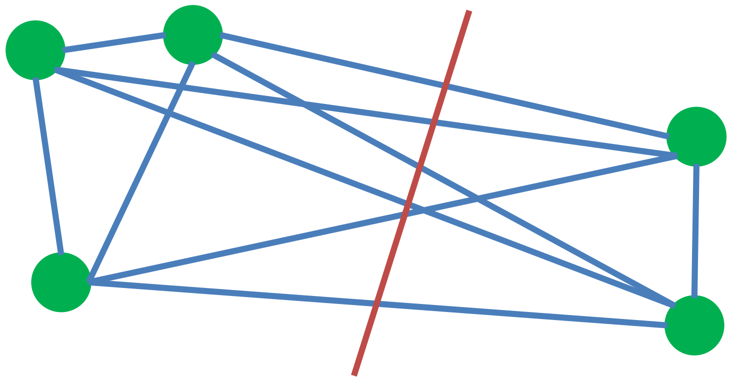
Clustering as graph cut

- Describe the pairwise distance via a graph
  - Clustering can be obtained via graph cut



Clustering as graph cut

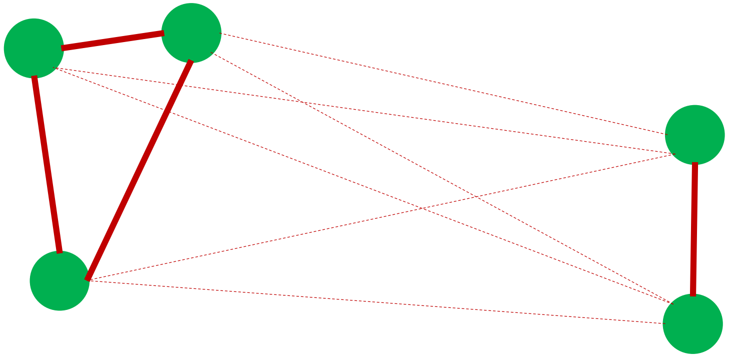
- Describe the pairwise distance via a graph
  - Clustering can be obtained via graph cut



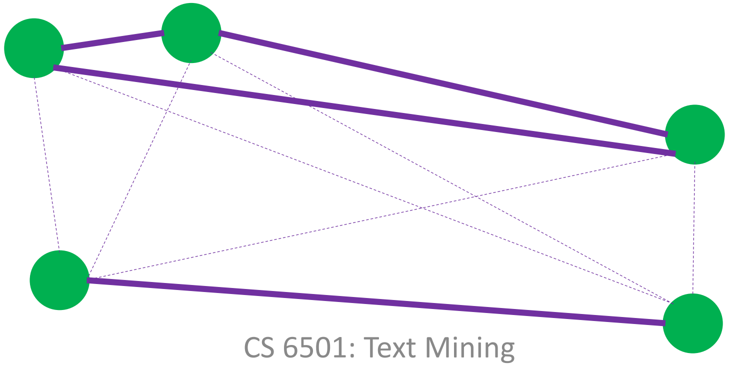
Clustering as graph cut

- Describe the pairwise distance via a graph
  - Clustering can be obtained via graph cut

Cut by class label



Cut by cluster label



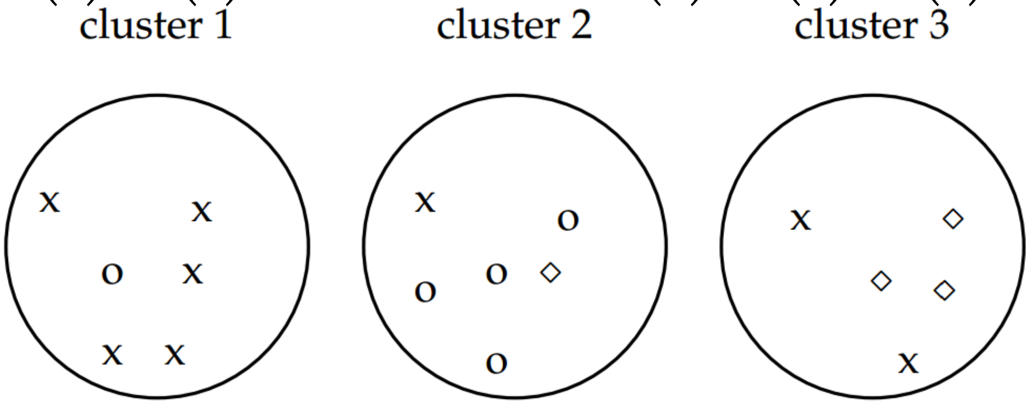
Recap: external validation

- Given class label Ω on each instance
  - Rand index

	$w_i = \mathrm{w_j}$	$w_i \neq \mathrm{w_j}$
$c_i = c_j$	20	20
$c_i \neq c_j$	24	72

$$TP + FP = \binom{6}{2} + \binom{6}{2} + \binom{5}{2} = 40$$

$$TP = \binom{5}{2} + \binom{4}{2} + \binom{3}{2} + \binom{2}{2} = 20$$



k-means Clustering

Hongning Wang

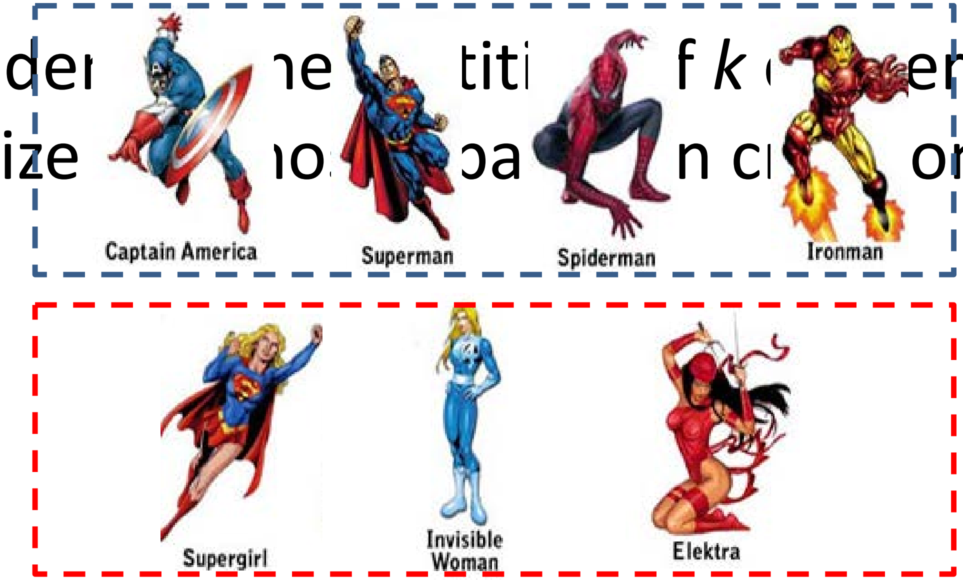
CS@UVa

Today's lecture

- *k*-means clustering
  - A typical partitional clustering algorithm
  - Convergence property
    - Expectation Maximization algorithm
  - Gaussian mixture model

Partitional clustering algorithms

- Partition instances into exactly *k* non-overlapping clusters
  - Flat structure clustering
  - Users need to specify the cluster size *k*
  - Task: ide he titi f *k* ers that optimize io pa n cr on



Partitional clustering algorithms

- Partition instances into exactly *k* non-overlapping clusters
  - Typical criterion
    - $\max \sum_{i\neq \mathrm{j}} d(c_i, c_j) - C \sum_i \sigma_i$

Optimize this in an alternative way

Inter-cluster distance

Intra-cluster distance

  - Optimal solution: enumerate every possible partition of size *k* and return the one maximizes the criterion

Let’s approximate this!

Unfortunately, this is NP-hard!

k-means algorithm

Input: cluster size *k*, instances $\{x_i\}_{i=1}^N$, distance metric $d(\cdot,\cdot)$

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

1. Initialize *k* cluster centroids $\{c_i\}_{i=1}^k$ (randomly if no domain knowledge is available)
2. Repeat until no instance changes its cluster membership:
   - Decide the cluster membership of instances by assigning them to the nearest cluster centroid

     $$z_i = argmin_k d(c_k, x_i)$$

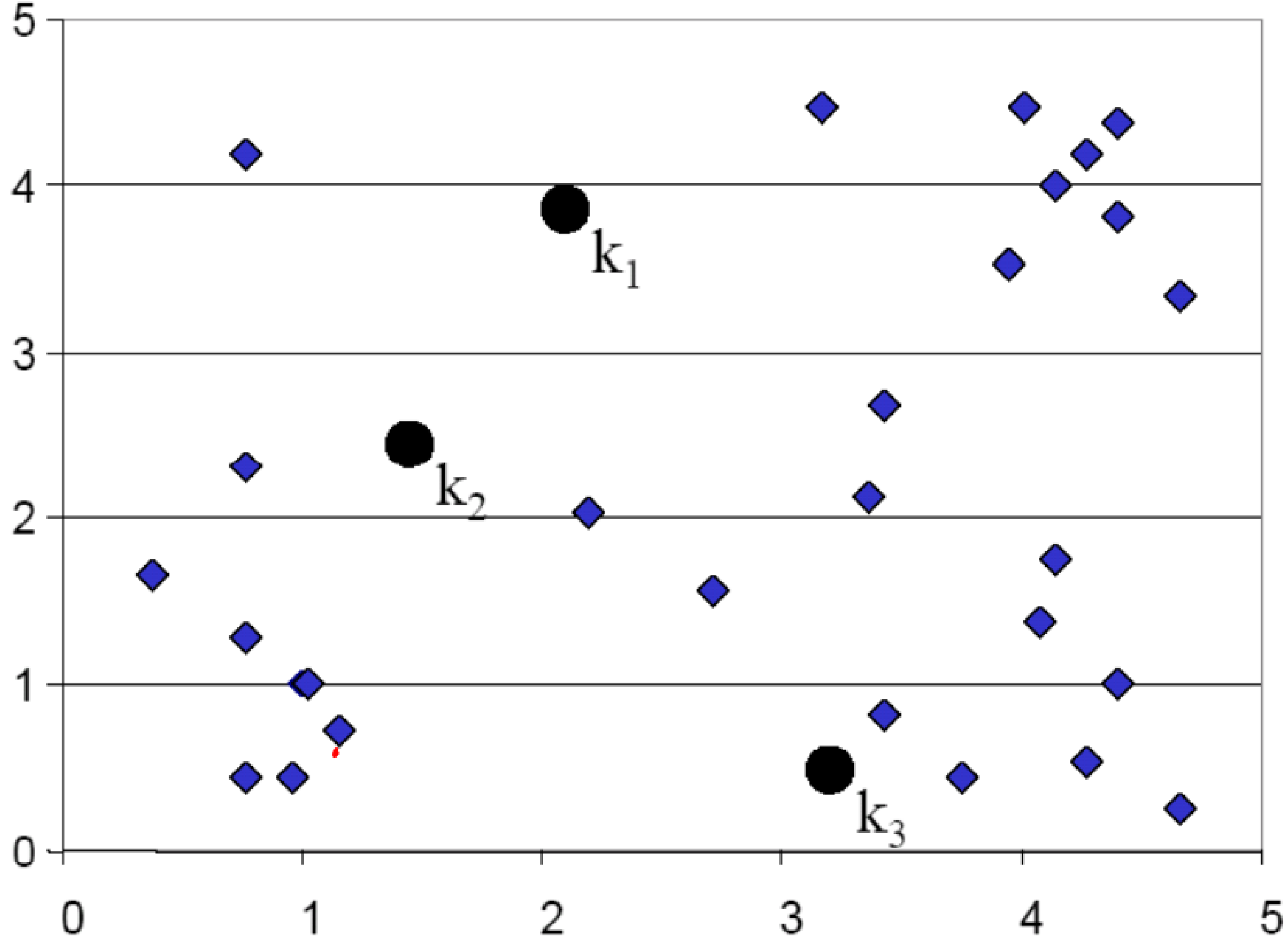
     Minimize intra distance

   - Update the *k* cluster centroids based on the assigned cluster membership

     $$c_k = \frac{\sum_i \delta(z_i = c_k) x_i}{\sum_i \delta(z_i = c_k)}$$

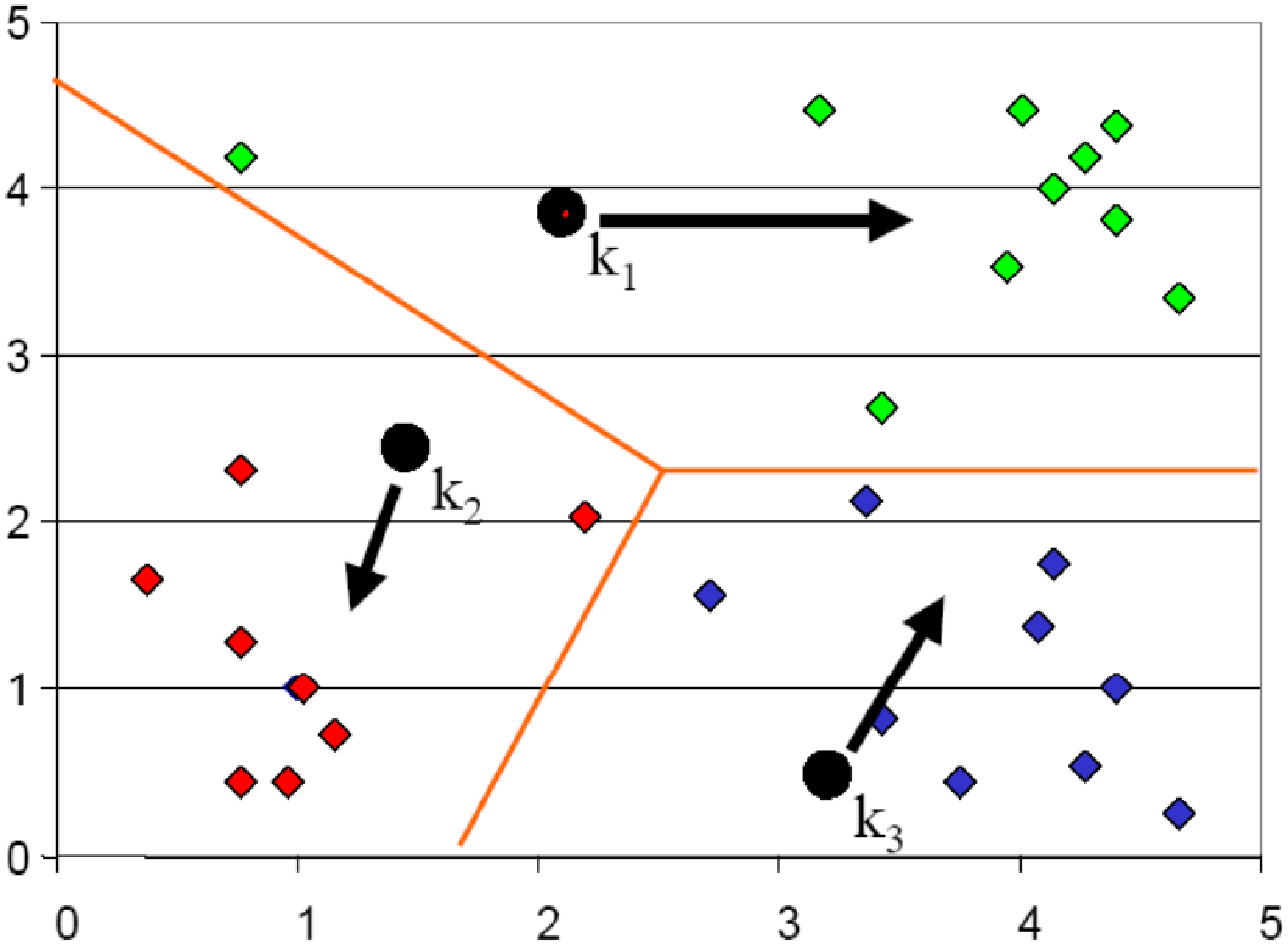
     Maximize inter distance

k-means illustration

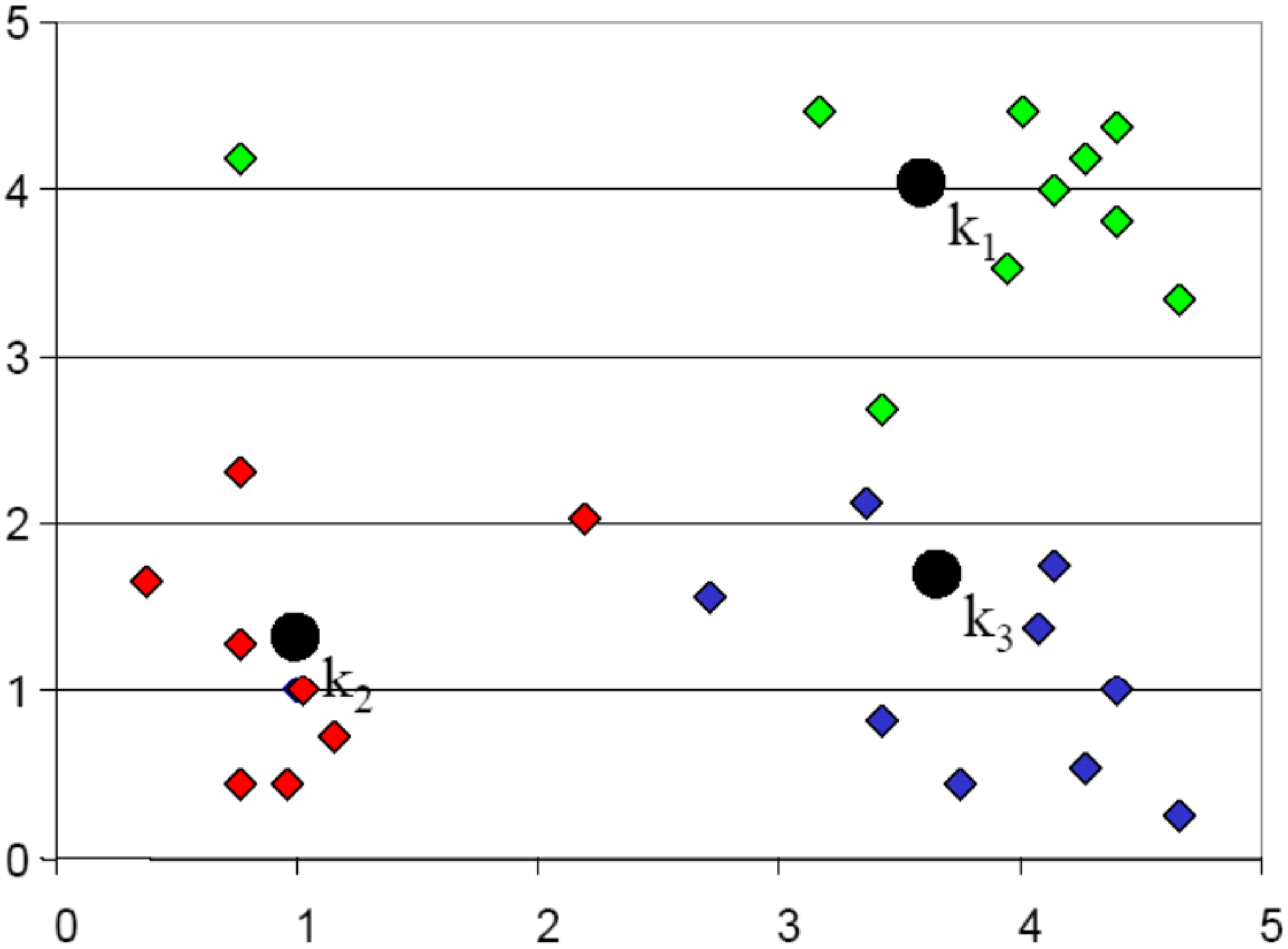


k-means illustration

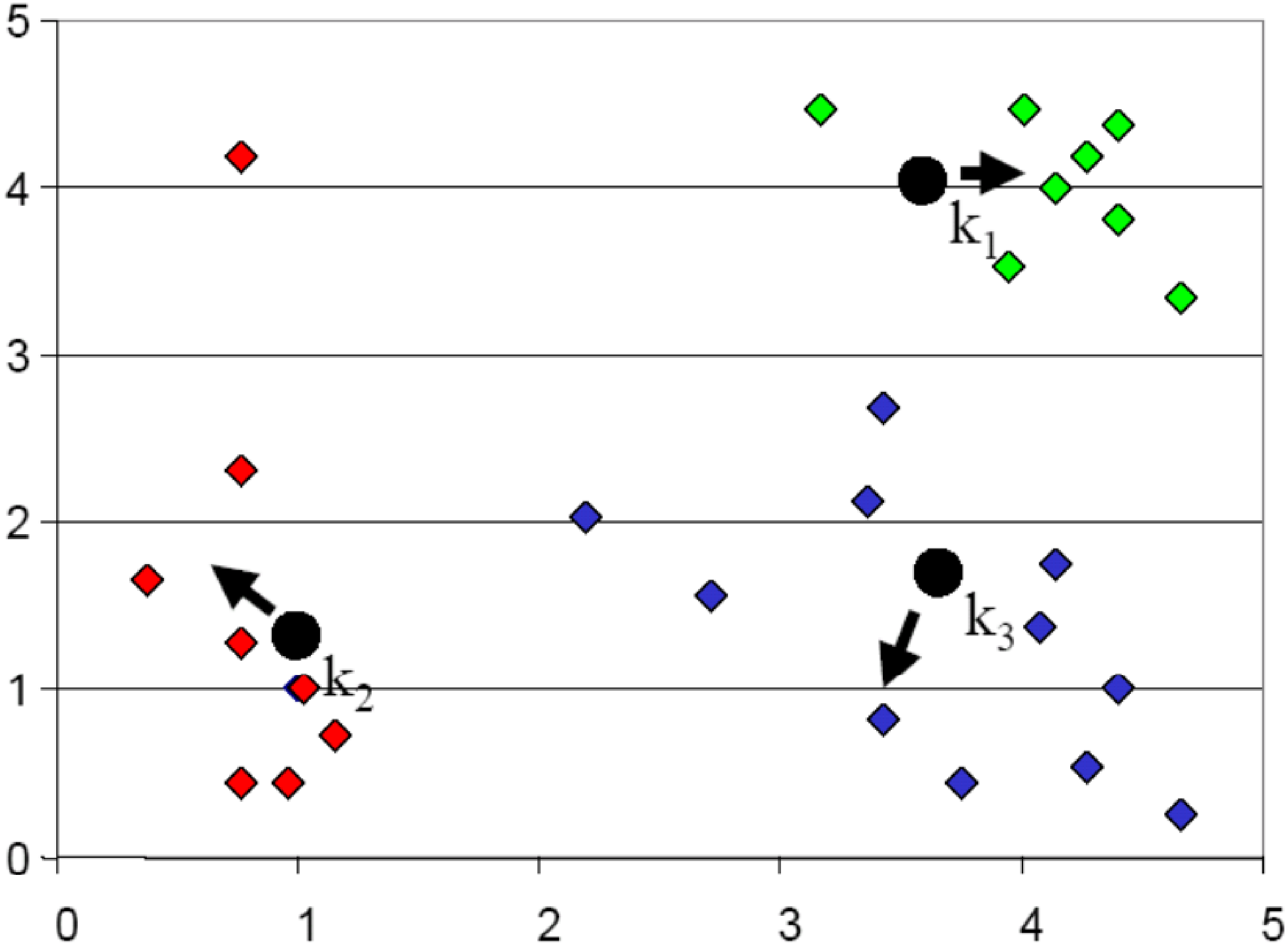
Voronoi diagram



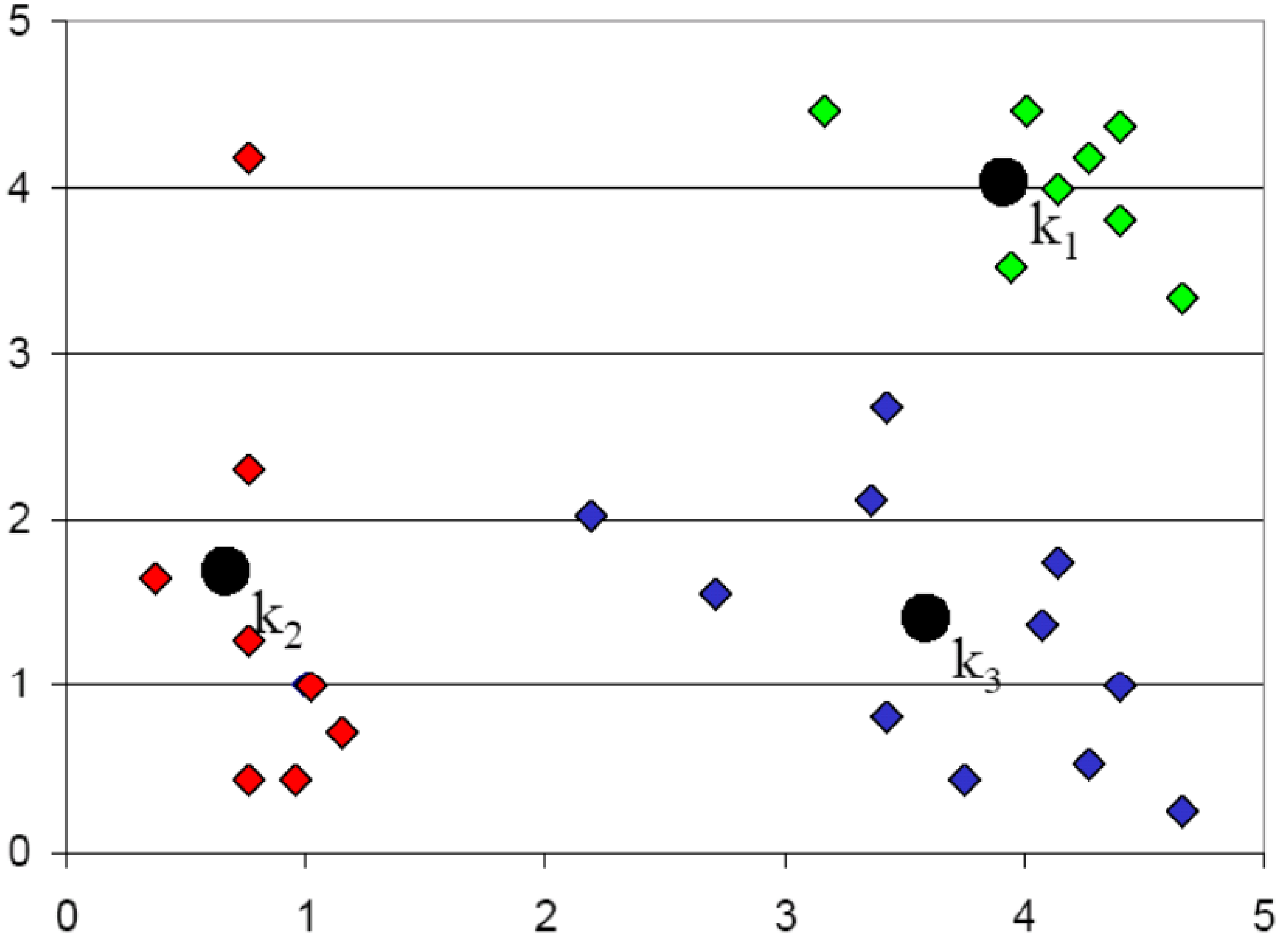
k-means illustration



k-means illustration



k-means illustration



Complexity analysis

- Decide cluster membership
  - $O(kn)$
- Compute cluster centroid
  - $O(n)$
- Assume *k*-means stops after l iterations
  - $O(knl)$

Don't forget the complexity of distance computation, e.g., $O(V)$ for Euclidean distance

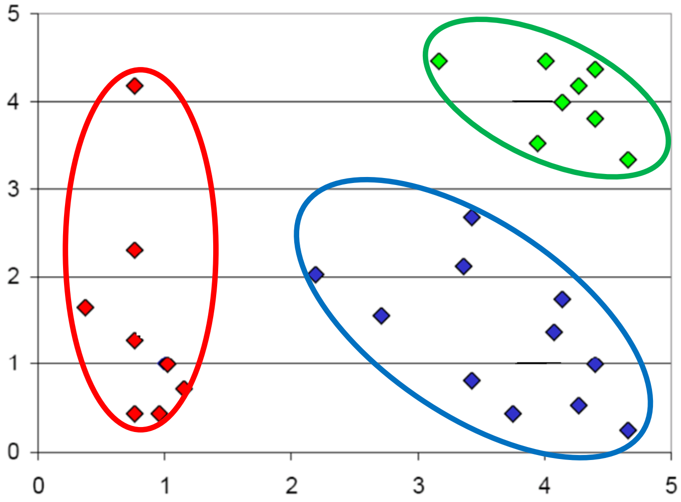
Convergence property

- Why will *k*-means stop?
  - Answer: it is a special version of Expectation Maximization (EM) algorithm, and EM is guaranteed to converge
  - However, it is only guaranteed to converge to local optimal, since *k*-means (EM) is a greedy algorithm

Probabilistic interpretation of clustering

- The density model of $p(x)$ is multi-modal
- Each mode represents a sub-population
  - – E.g., unimodal Gaussian for each group

$p(x|z=1)$



$p(x|z=2)$

$p(x|z=3)$

Mixture model

$$p(x) = \sum_z p(x|z)p(z)$$

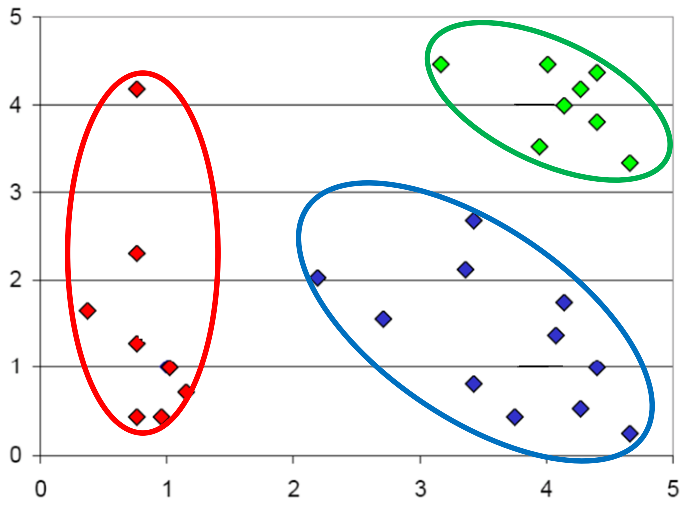
Unimodal distribution

Mixing proportion

Probabilistic interpretation of clustering

- If z is known for every x
  - Estimating $p(z)$ and $p(x|z)$ is easy
    - Maximum likelihood estimation
    - This is Naïve Bayes

$p(x|z=1)$



$p(x|z=2)$

$p(x|z=3)$

Mixture model

$$p(x) = \sum_z p(x|z)p(z)$$

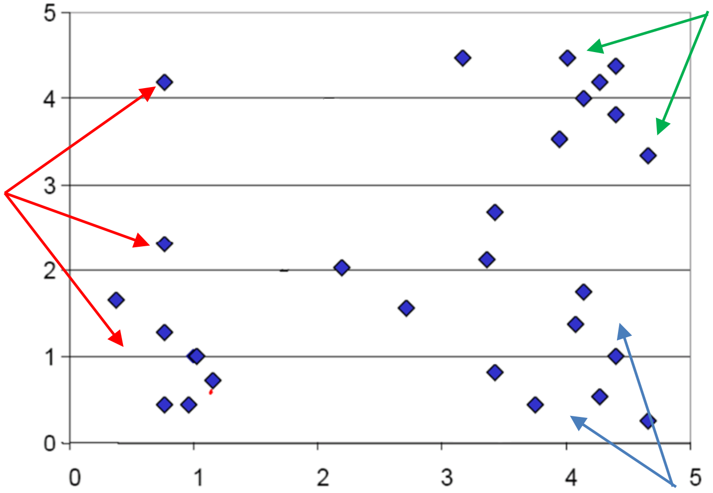
Unimodal distribution

Mixing proportion

Probabilistic interpretation of clustering

- But z is unknown for all x
  - Estimating $p(z)$ and $p(x|z)$ is generally hard
    - $\max_{\alpha,\beta} \sum_i \log \sum_{z_i} p(x_i|z_i,\beta)p(z_i|\alpha)$
  - Appeal to the Expectation Maximization algorithm

Usually a constrained optimization problem



$p(x|z=1)?$

$p(x|z=2)?$

$p(x|z=3)?$

Mixture model

$$p(x) = \sum_z p(x|z)p(z)$$

Unimodal distribution

Mixing proportion

Introduction to EM

- Parameter estimation
  - All data is observable
    - Maximum likelihood estimator
    - Optimize the analytic form of $L(\theta) = \log p(X|\theta)$
  - Missing/unobservable data
    - Data: X (observed) + Z (hidden)
    - Likelihood: $L(\theta) = \log \sum_z p(X, Z|\theta)$
    - Approximate it!

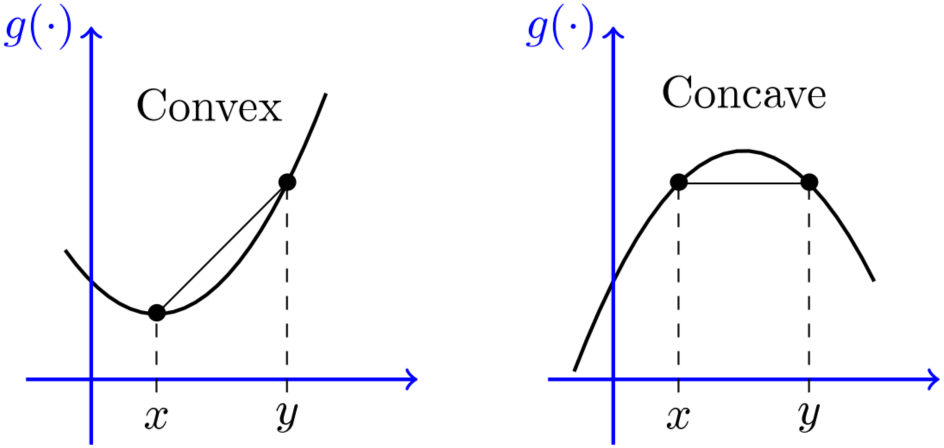
E.g. cluster membership

Most of cases are intractable

Background knowledge

- Jensen's inequality
  - For any convex function $f(x)$ and positive weights λ,

$$f\left(\sum_i \lambda_i x_i\right) \le \sum_i \lambda_i f(x_i) \qquad \sum_i \lambda_i = 1$$



Expectation Maximization

- Maximize data likelihood function by pushing the lower bound

Proposal distributions for Z

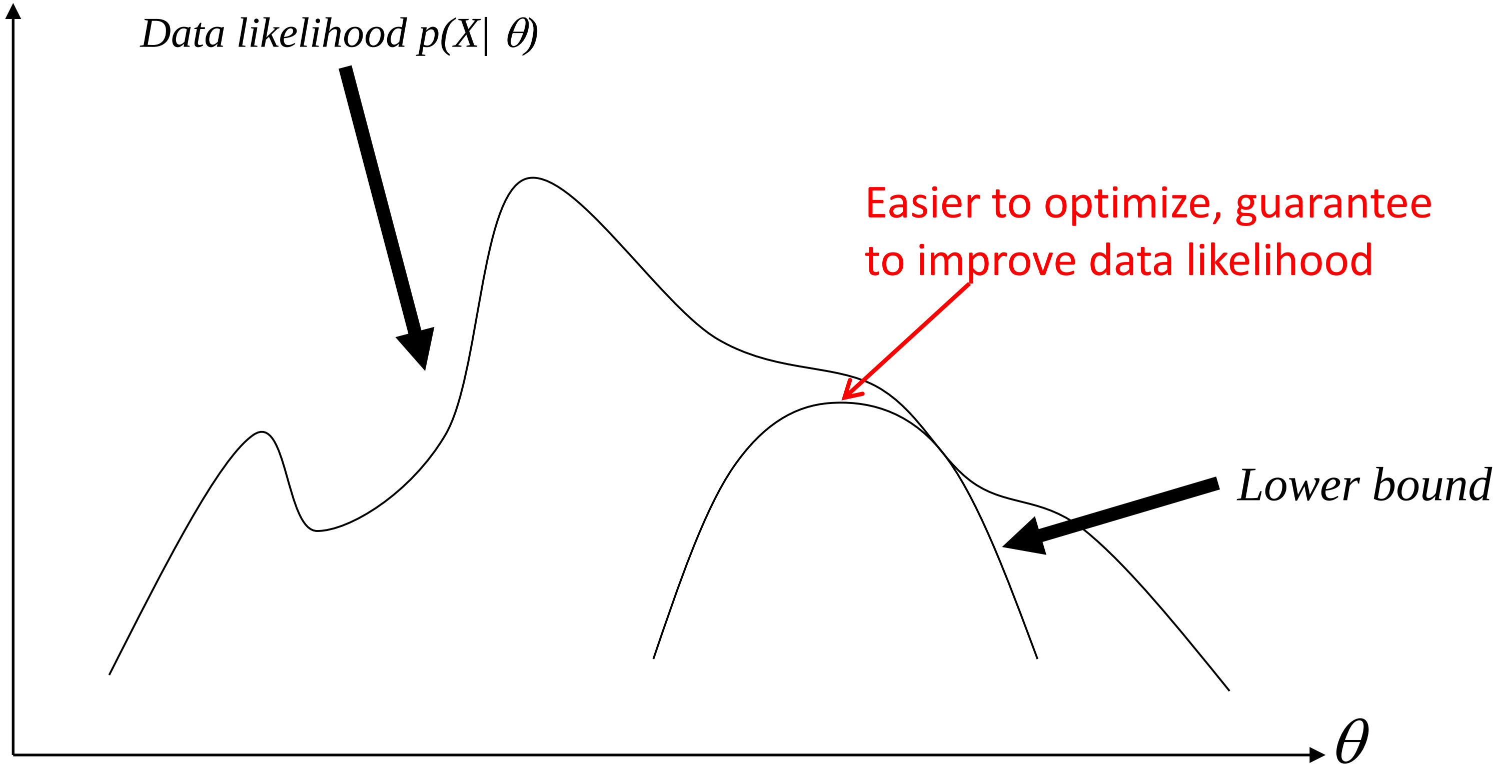
$$-L(\theta) = \log \sum_Z p(X, \mathrm{Z}|\theta) = \log \sum_Z \frac{q(Z)\, p(X, \mathrm{Z}|\theta)}{q(Z)}$$

Jensen's inequality
$f(E[x]) \geq E[f(x)]$

$$\geq \underbrace{\sum_Z q(Z) \log p(X, Z|\theta) - \sum_Z q(Z) \log q(Z)}_{\textit{Lower bound!}}$$

Components we need to tune when optimizing $\boldsymbol{L(\theta)}$: $\boldsymbol{q(Z)}$ and $\boldsymbol{\theta}$!

Intuitive understanding of EM


Data likelihood p(X| θ)
Easier to optimize, guarantee to improve data likelihood
Lower bound
θ

Expectation Maximization (cont)

- Optimize the lower bound w.r.t. $q(Z)$
  - $L(\theta) \geq \sum_Z q(Z)\, log\, p(X,Z|\theta) - \sum_Z q(Z)\, log\, q(Z)$

$$= \sum_Z \underline{q(Z)[\log p(Z|X,\theta)} + \underline{\log p(X|\theta)}] - \sum_Z \underline{q(Z)\log q(Z)}$$

$$= \sum_Z q(Z)\log\frac{p(Z|X,\theta)}{q(Z)} + \log p(X|\theta)$$

negative KL-divergence between $q(Z)$ and $p(Z|X,\theta)$

Constant with respect to $q(Z)$

$$KL(P||Q) = \int P(x)\log\frac{P(x)}{Q(x)}dx$$

Expectation Maximization (cont)

- Optimize the lower bound w.r.t. $q(Z)$
  - $L(\theta) \geq -KL\big(q(Z)||p(Z|X,\theta)\big) + L(\theta)$
  - KL-divergence is non-negative, and equals to zero i.f.f. $q(Z) = p(Z|X,\theta)$
  - A step further: when $q(Z) = p(Z|X,\theta)$, we will get $L(\theta) \geq L(\theta)$, i.e., the lower bound is tight!
  - Other choice of $q(Z)$ cannot lead to this tight bound, but might reduce computational complexity
  - **Note**: calculation of $q(Z)$ is based on current θ

Expectation Maximization (cont)

- Optimize the lower bound w.r.t. $q(Z)$
  - Optimal solution: $q(Z) = p(Z|X,\theta^t)$

Posterior distribution of Z given current model θ^t

In k-means: this corresponds to assigning instance x_i to its closest cluster centroid c_k

$$z_i = argmin_k d(c_k, x_i)$$

Expectation Maximization (cont)

- Optimize the lower bound w.r.t. θ
  - $L(\theta) \geq \sum_Z p(Z|X,\theta^t) \log p(X,Z|\theta) - \xcancel{\sum_Z p(Z|X,\theta^t) \log p(Z|X,\theta^t)}$ ⟵ Constant w.r.t. θ
  - $\theta^{t+1} = argmax_\theta \sum_Z p(Z|X,\theta^t)\, log\, p(X,Z|\theta)$

$$= argmax_\theta E_{Z|X,\theta^t}[\log p(X,Z|\theta)]$$

↑

Expectation of complete data likelihood

In k-means, we are <u>not</u> computing the expectation, but the most probable configuration, and then $c_k = \frac{\sum_i \delta(z_i = c_k) x_i}{\sum_i \delta(z_i = c_k)}$

Expectation Maximization

- EM tries to iteratively maximize likelihood
  - "Complete" likelihood: $L^c(\theta) = \log p(X, \mathrm{Z}|\theta)$
  - Starting from an initial guess $\theta^{(0)}$,
    1. **E-step**: compute the expectation of the complete likelihood

$$Q(\theta; \theta^t) = \mathrm{E}_{Z|X,\theta^t}[L^c(\theta)] = \sum_Z \underline{p(Z|X, \theta^t)} \log \mathrm{p}(\mathrm{X}, \mathrm{Z}|\theta)$$

    2. **M-step**: compute $\theta^{(t+1)}$ by maximizing the Q-function

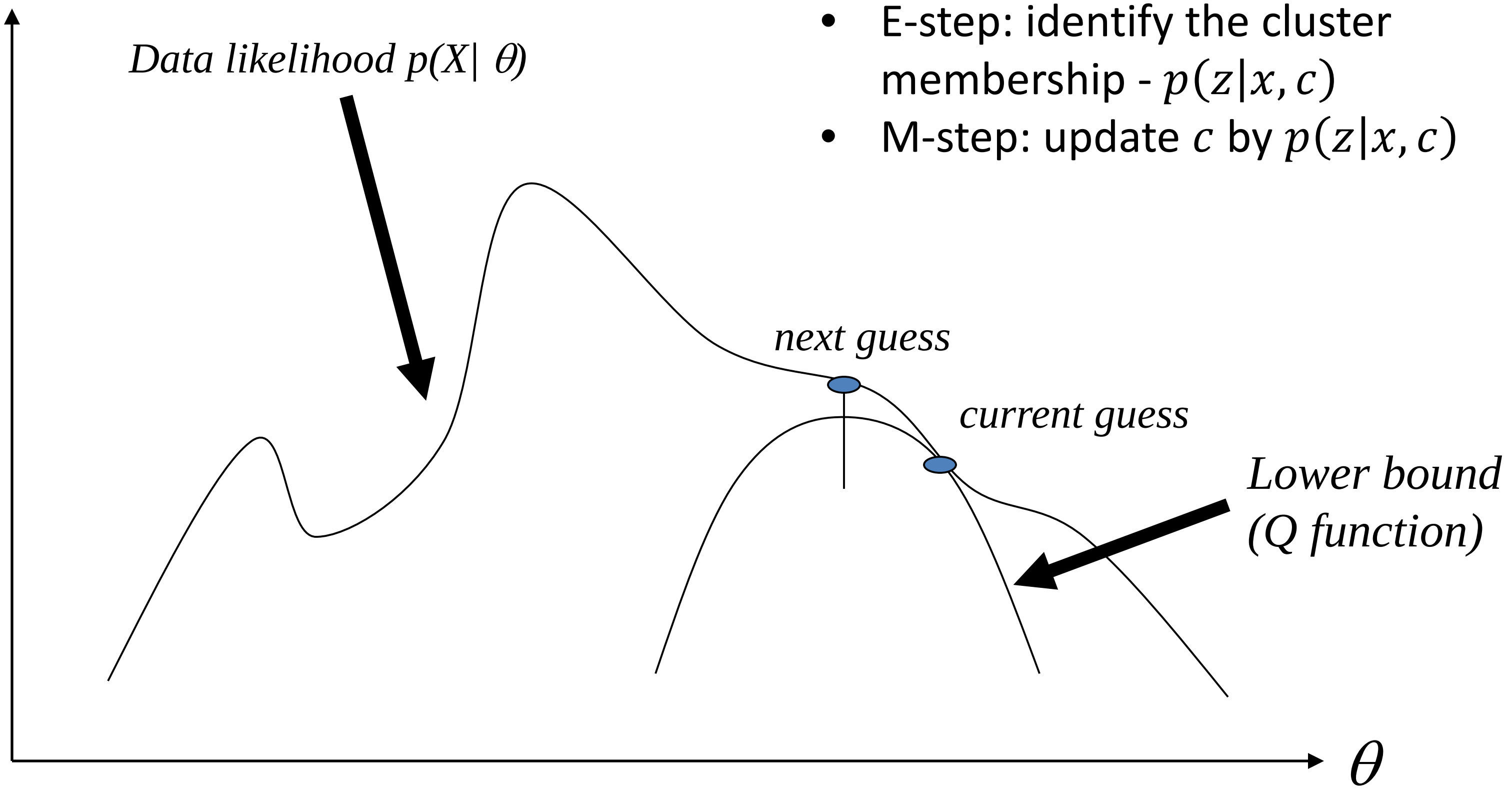
$$\theta^{t+1} = argmax_\theta Q(\theta; \theta^t)$$

Key step!

An intuitive understanding of EM

In *k*-means

- E-step: identify the cluster membership - $p(z|x,c)$
- M-step: update c by $p(z|x,c)$



E-step = computing the lower bound
M-step = maximizing the lower bound

Convergence guarantee

- Proof of EM

$$\log p(X|\theta) = \log p(Z,X|\theta) - \log p(Z|X,\theta)$$

Taking expectation with respect to $p(Z|X,\theta^t)$ of both sides:

$$\log p(X|\theta) = \sum_Z p(Z|X,\theta^t)\log p(Z,X|\theta) - \sum_Z p(Z|X,\theta^t)\log p(Z|X,\theta)$$

$$= Q(\theta;\theta^t) + \underline{H(\theta;\theta^t)} \longleftarrow \text{Cross-entropy}$$

Then the change of log data likelihood between EM iteration is:

$$\log p(X|\theta) - \log p(X|\theta^t) = Q(\theta;\theta^t) + H(\theta;\theta^t) - Q(\theta^t;\theta^t) - H(\theta^t;\theta^t)$$

By Jensen's inequality, we know $H(\theta;\theta^t) \geq H(\theta^t;\theta^t)$, that means

$$\log p(X|\theta) - \log p(X|\theta^t) \geq Q(\theta;\theta^t) - Q(\theta^t;\theta^t) \underline{\geq 0}$$

M-step guarantee this

What is not guaranteed

- Global optimal is not guaranteed!
  - Likelihood: $L(\theta) = \log \sum_Z p(X, \mathrm{Z}|\theta)$ is non-convex in most of case
  - EM boils down to a greedy algorithm
    - Alternative ascent
- Generalized EM
  - E-step: $\hat{q}(Z) = \mathrm{argmin}_{q(Z)} KL\big(q(Z)||p(Z|X, \theta^t)\big)$
  - M-step: choose θ that improves $Q(\theta; \theta^t)$

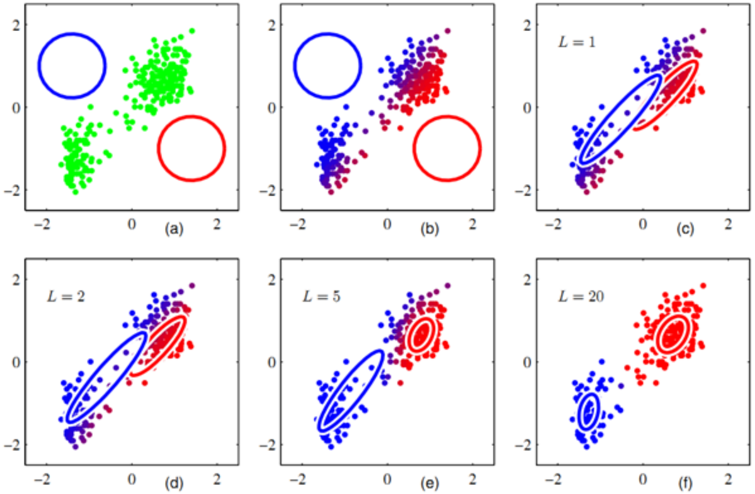
k-means v.s. Gaussian Mixture

- If we use Euclidean distance in *k*-means
  - We have explicitly assumed $p(x|z)$ is Gaussian
  - Gaussian Mixture Model (GMM)
    - $p(x|z) = N(\mu_z, \Sigma_z)$
    - $p(z) = \alpha_z$ ← Multinomial

$$P(x|z) = \frac{1}{\sqrt{(2\pi)^k \Sigma_z}} e^{-\frac{1}{2}(x-\mu_z)^T \Sigma_z^{-1}(x-\mu_z)}$$

$$\frac{1}{\sqrt{2}} e^{-\frac{(x-\mu_z)^T(x-\mu_z)}{2}}$$

We do not consider cluster size in *k*-means

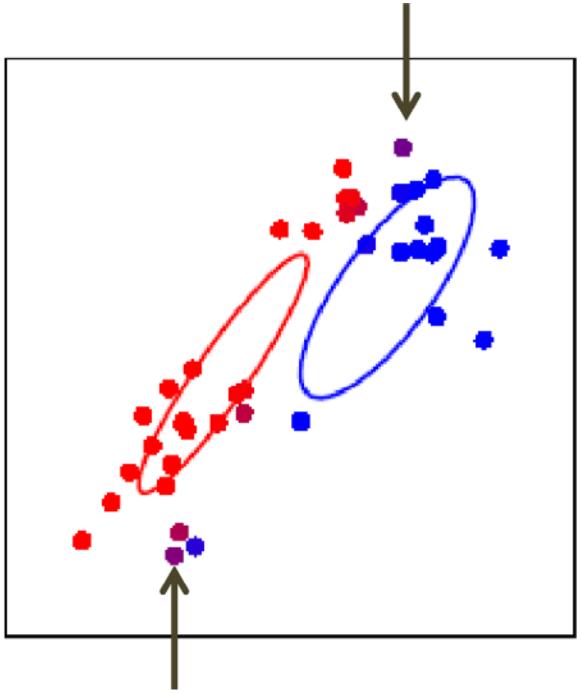


In *k*-means, we assume equal variance across clusters, so we don't need to estimate them

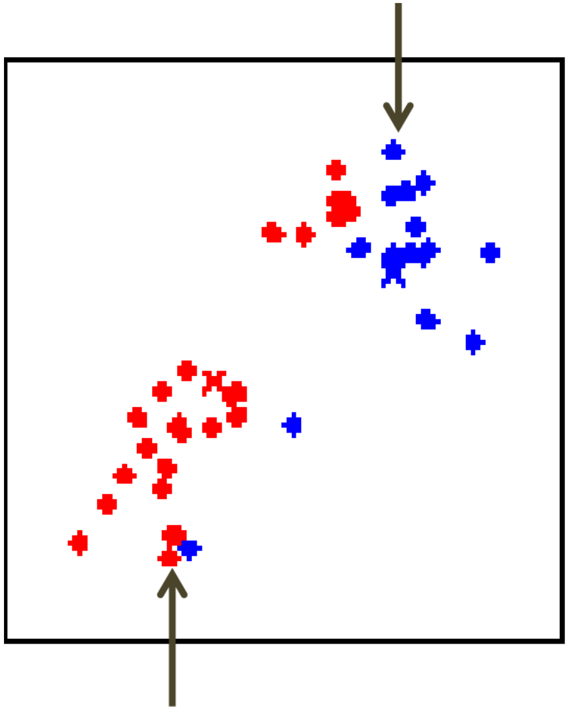
k-means v.s. Gaussian Mixture

- Soft v.s., hard posterior assignment

GMM



k-means



k-means in practice

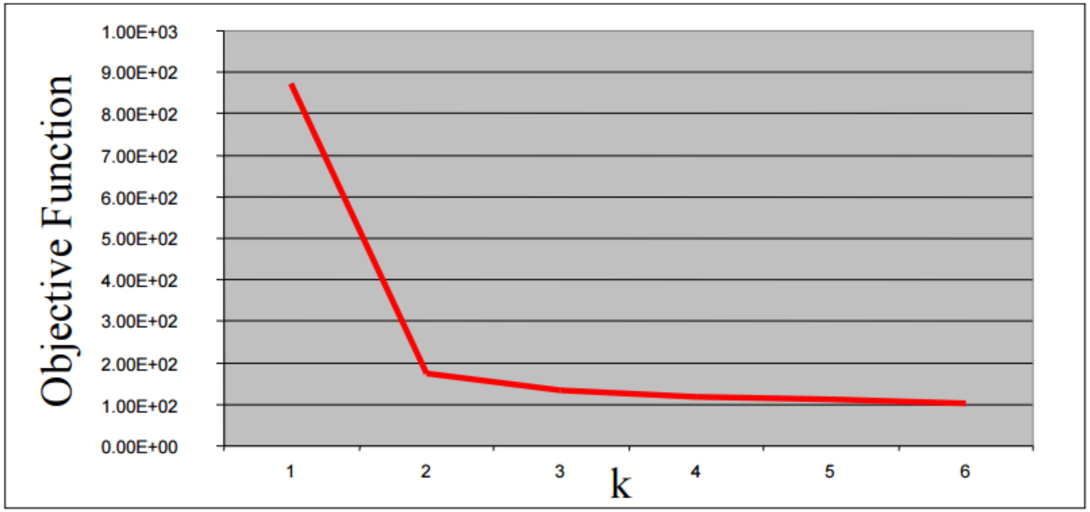
- Extremely fast and scalable
  - One of the most popularly used clustering methods
    - Top 10 data mining algorithms – ICDM 2006
  - Can be easily parallelized
    - Map-Reduce implementation
      - Mapper: assign each instance to its closest centroid
      - Reducer: update centroid based on the cluster membership
  - Sensitive to initialization
    - Prone to local optimal

Better initialization: *k*-means++

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

How to determine k

- Vary k to optimize clustering criterion
  - Internal v.s. external validation
  - Cross validation
    - Abrupt change in objective function



How to determine k

- Vary k to optimize clustering criterion
  - Internal v.s. external validation
  - Cross validation
    - Abrupt change in objective function
    - Model selection criterion – penalizing too many clusters
      - AIC, BIC

What you should know

- *k*-means algorithm
  - An alternative greedy algorithm
  - Convergence guarantee
    - EM algorithm
  - Hard clustering v.s., soft clustering
    - *k*-means v.s., GMM

Today's reading

- Introduction to Information Retrieval
  - Chapter 16: Flat clustering
    - 16.4 *k*-means
    - 16.5 Model-based clustering