CMPT 825
Natural Language Processing

Anoop Sarkar

http://www.cs.sfu.ca/~anoop

Clustering

- Clustering is unsupervised classification
- $C = \{c_1, \ldots, c_n\}$ classes, put each item x into one of these classes
- Classification depends on training data
- Clustering is based on the idea that we can collect items in the test data into n groups
- Cluster so that: similar items within a group, dissimilar between groups

Defining Similarity

- Consider items *X*, *Y* as binary vectors (e.g.g documents represented as a bag of words)
- matching coefficient $|X \cap Y|$
- Dice coefficient $\frac{2|X \cap Y|}{|X|+|Y|}$
- Jaccard (Tanimoto) coeff. $\frac{|X \cap Y|}{|X \cup Y|}$
- Overlap coeff. $\frac{|X \cap Y|}{min(|X|,|Y|)}$
- cosine $\frac{|X \cap Y|}{\sqrt{|X| \times |Y|}}$

Similarity

- Consider real-valued vectors $\vec{x}, \vec{y}$

$$\vec{x} = \langle x_1, \ldots, x_n \rangle$$

- length of a vector $$|\vec{x}| = \sqrt{\sum_{i=1}^{n} x_i^2}$$

- dot product $$\vec{x} \cdot \vec{y} = \sum_{i=1}^{n} x_i y_i$$

- cosine $$cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}||\vec{y}|}$$

Euclidian distance

- Euclidian distance (L_2 norm)

$$|\vec{x}-\vec{y}| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

- Ranking according to the Euclidian distance turns out to be the same as the cosine similarity
- L_p norm: $L_p(\vec{x},\vec{y}) = \left[\sum_{i=1}^{n}(x_i - y_i)^p\right]^{\frac{1}{p}}$

Distance (or *Dis*similarity)

- KL Divergence $D(p \,||\, q) = \sum_i p_i log \frac{p_i}{q_i}$
- information radius (IRad)

$$D(p \,||\, \frac{p+q}{2}) + D(q \,||\, \frac{p+q}{2})$$

- L_1 norm $\sum_i | p_i - q_i |$
- Similarity from distance: $sim(x,y) = \frac{1}{1+d(x,y)}$

Types of Clustering Algorithms

- Hierarchical vs. Flat clustering
  - hierarchical (agglomerative): similar to phylogenetic trees
  - flat clustering: K-means
- Soft vs. hard clustering
  - Hard: K-means
  - Soft: using the EM algorithm
- Proper distance metric is crucial to success

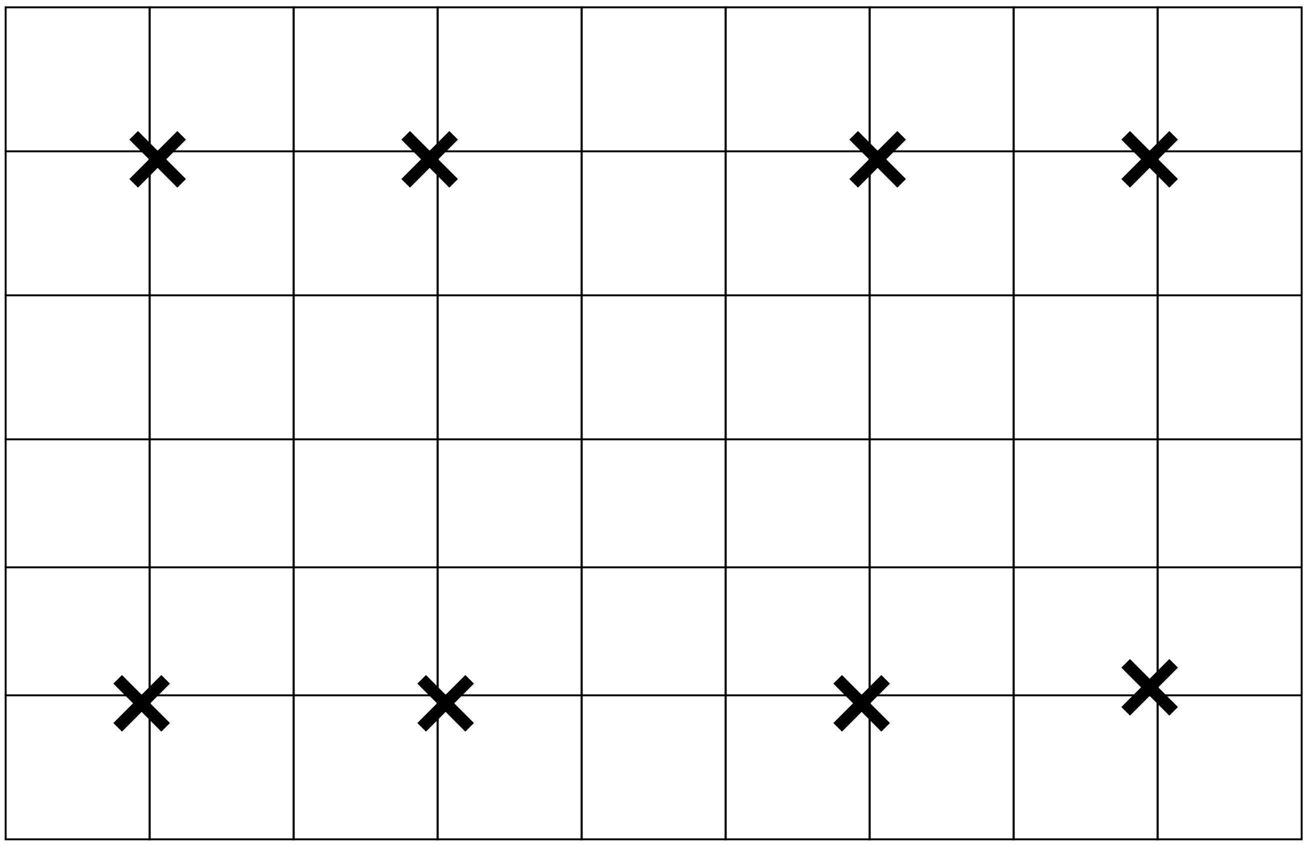
Hierarchical Clustering

- Bottom up h. clustering
  - Each item is allocated to a class
  - Iteratively combine two classes that are *most similar* to form a bigger class
- Top down h. clustering
  - Start with one big class with all the items
  - Pick the class that is most *incoherent* (with *least similar* items) and then split into two classes

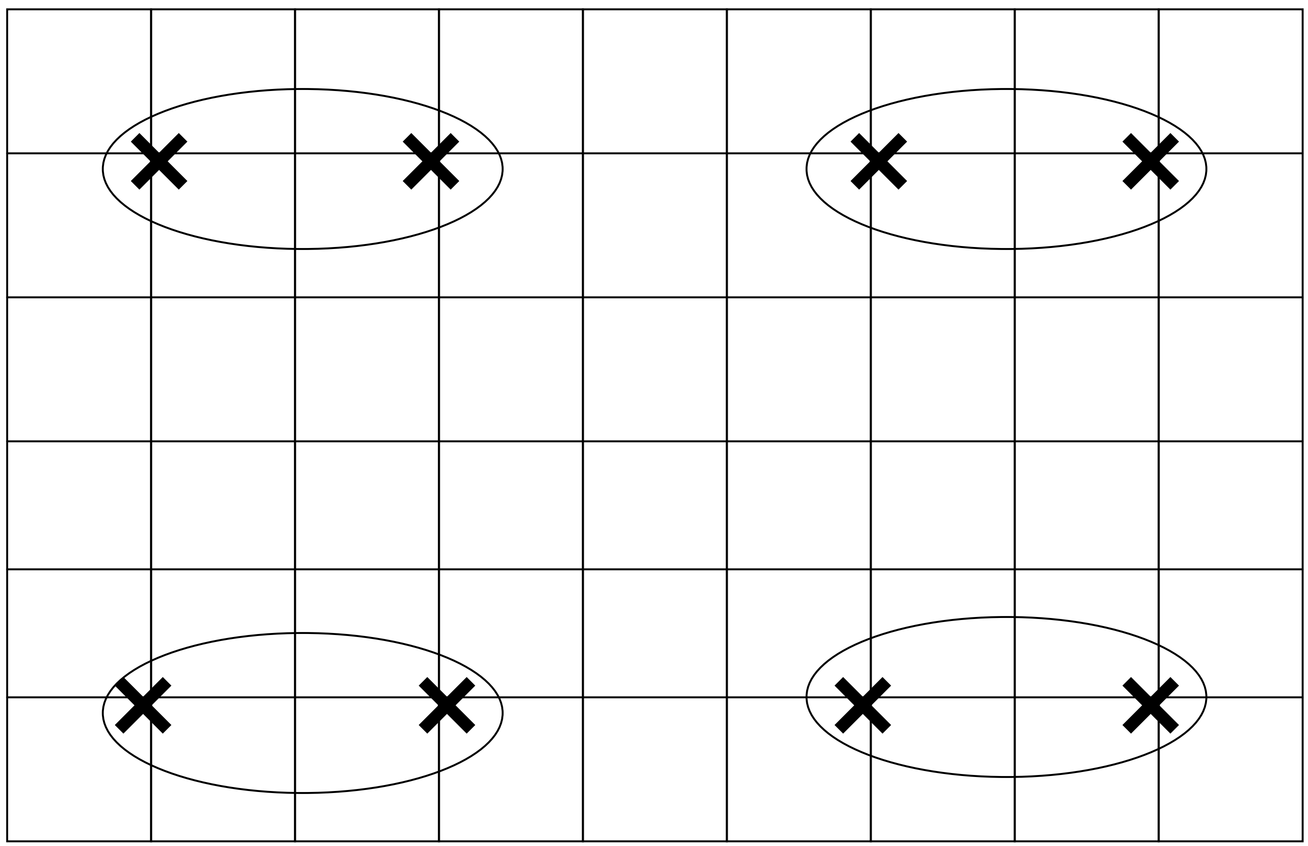
Class similarity

- We have similarity between two items, but for clustering we need similarity between *classes*
- Three methods:
  - single link: use most similar members
  - complete link: use least similar members
  - group average: average similarity between members, e.g. $cos(x,y)$

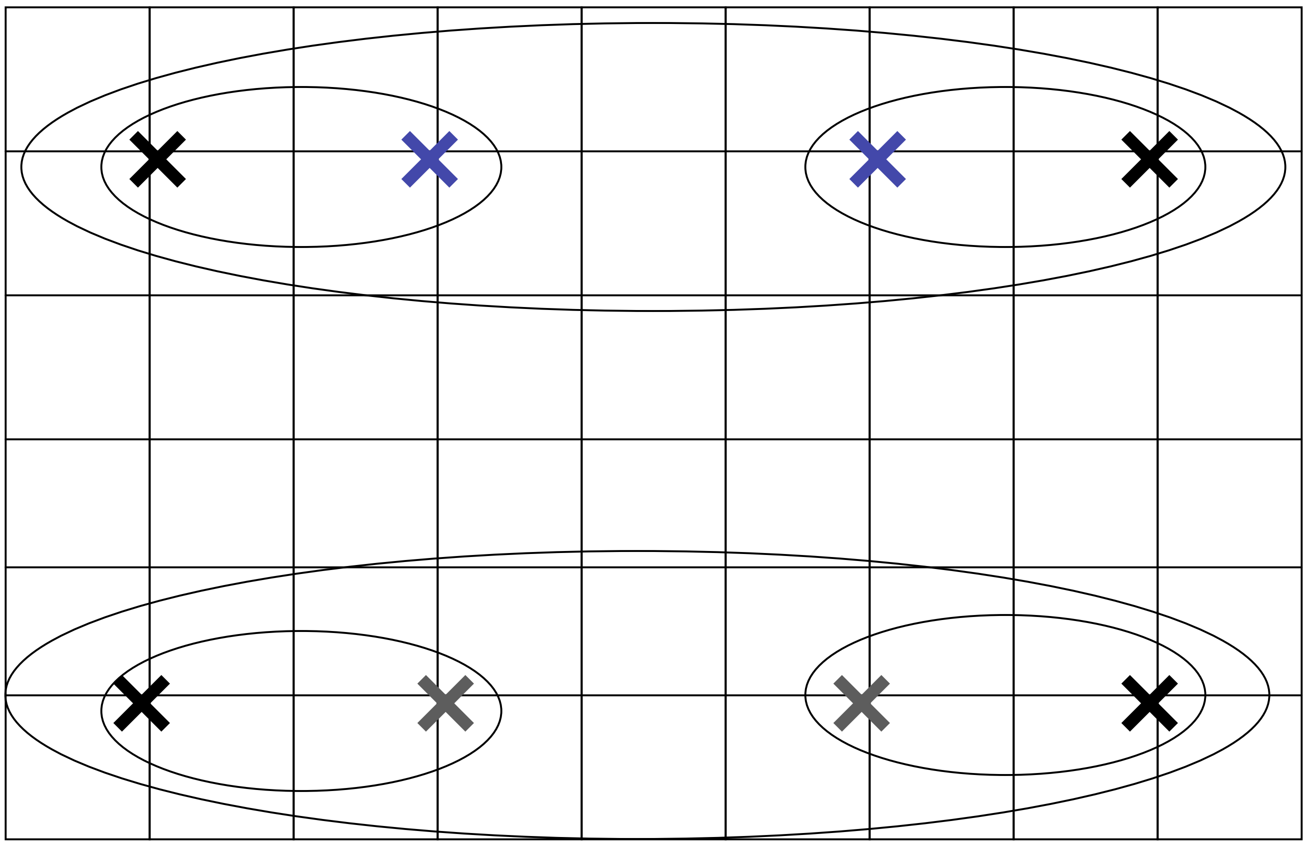
Single vs. Complete link



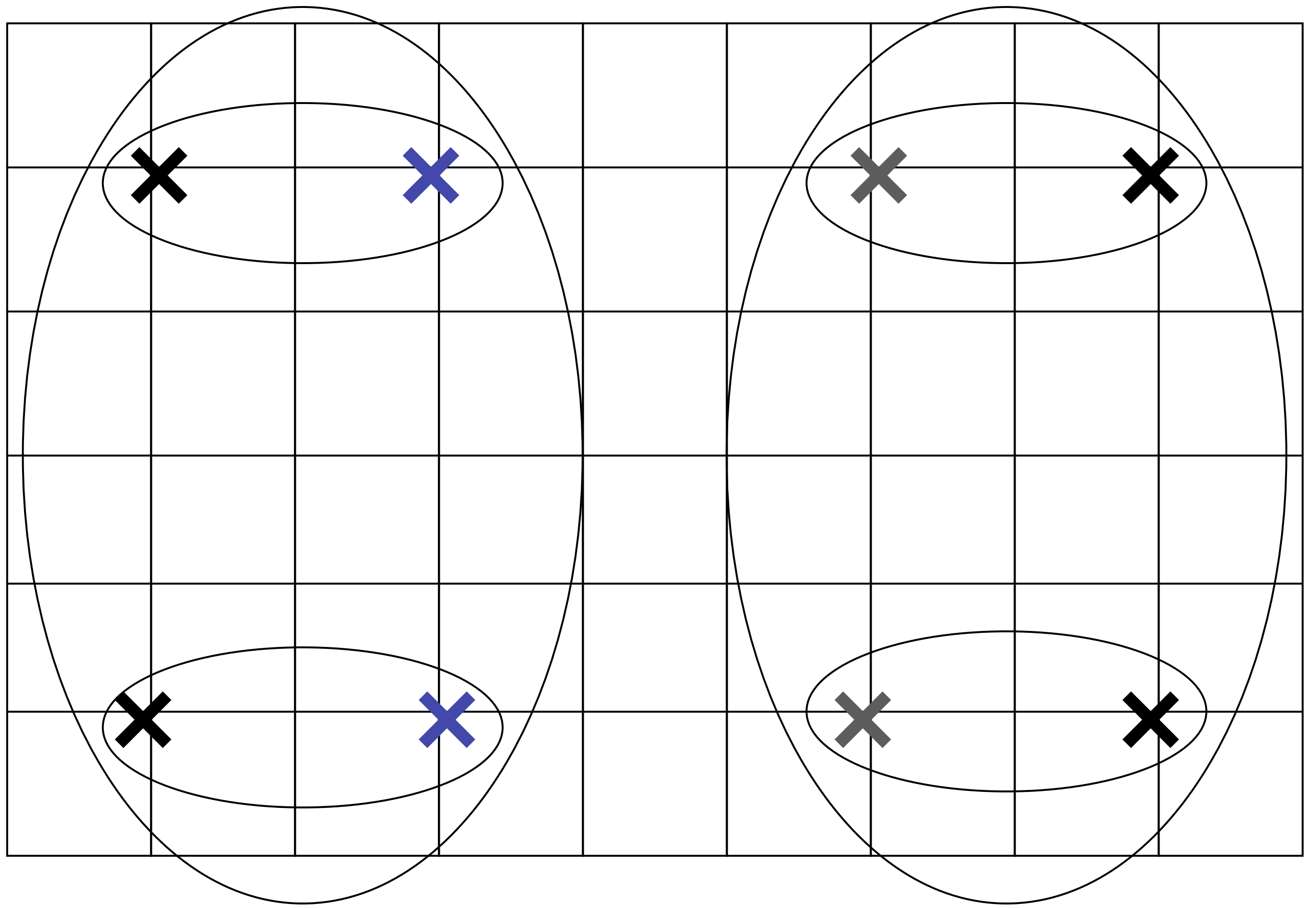
Single vs. Complete link



Single link



Complete link



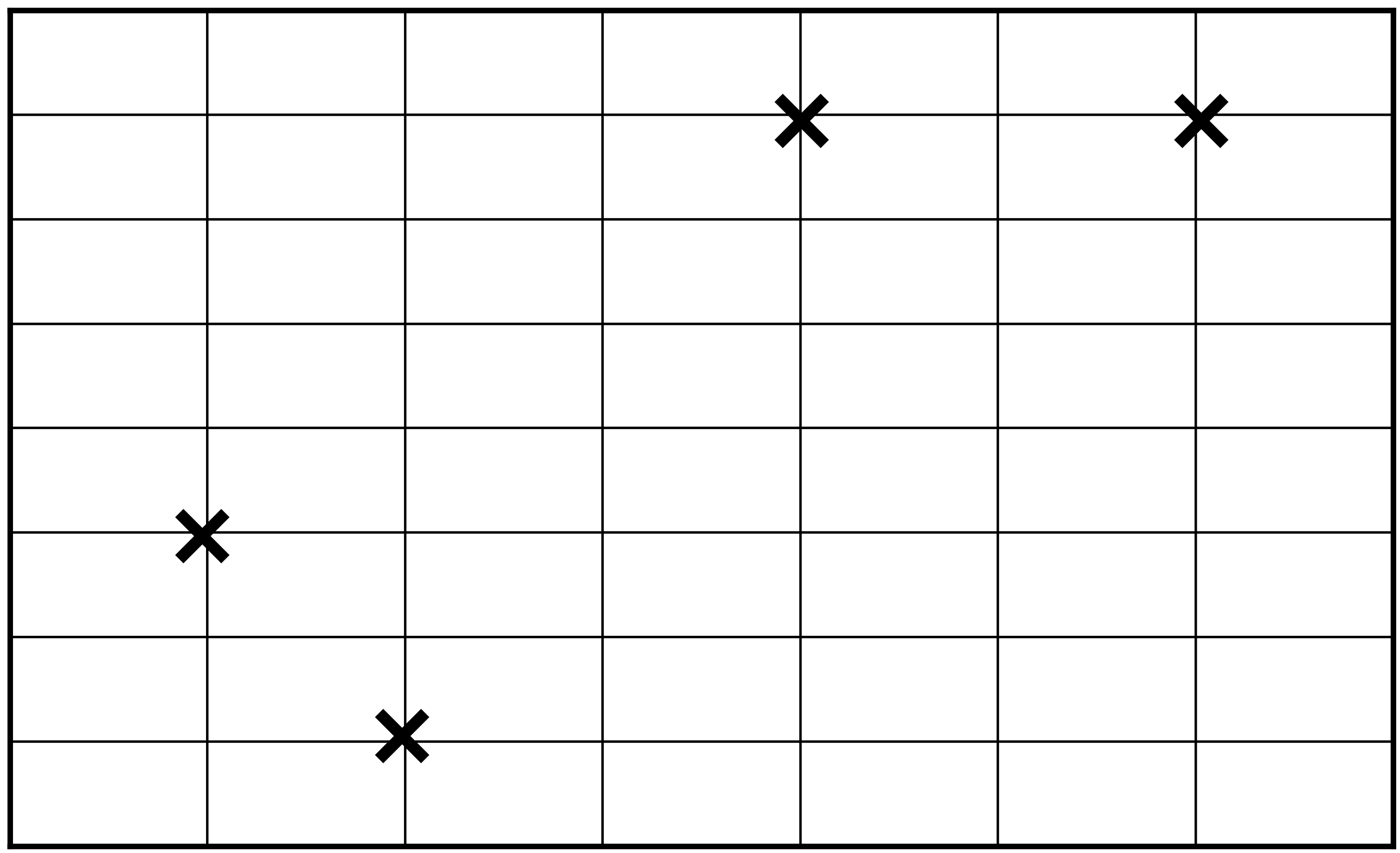
Non-hierarchical Clustering

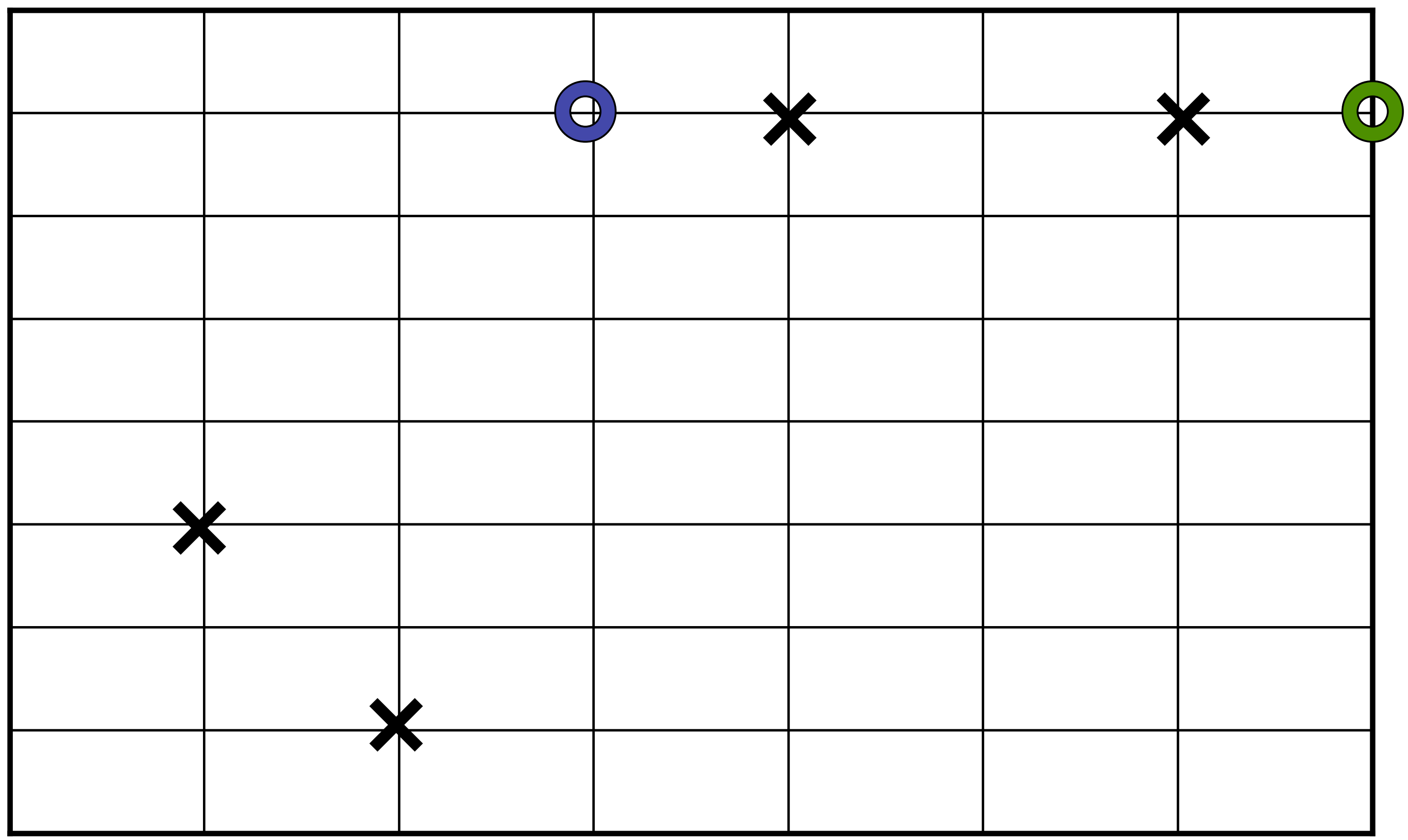
- K-means
  - hard clustering
  - computes centroid of a cluster
- EM algorithm
  - Consider sparse data in $P(w_2 \mid w_1)$
  - $P(w_2 \mid w_1) = P(c_i)\ P(c_i \mid w_1)\ P(w_2 \mid c_i)$
  - Compute $P(c_i \mid w_1)$ using the EM algorithm

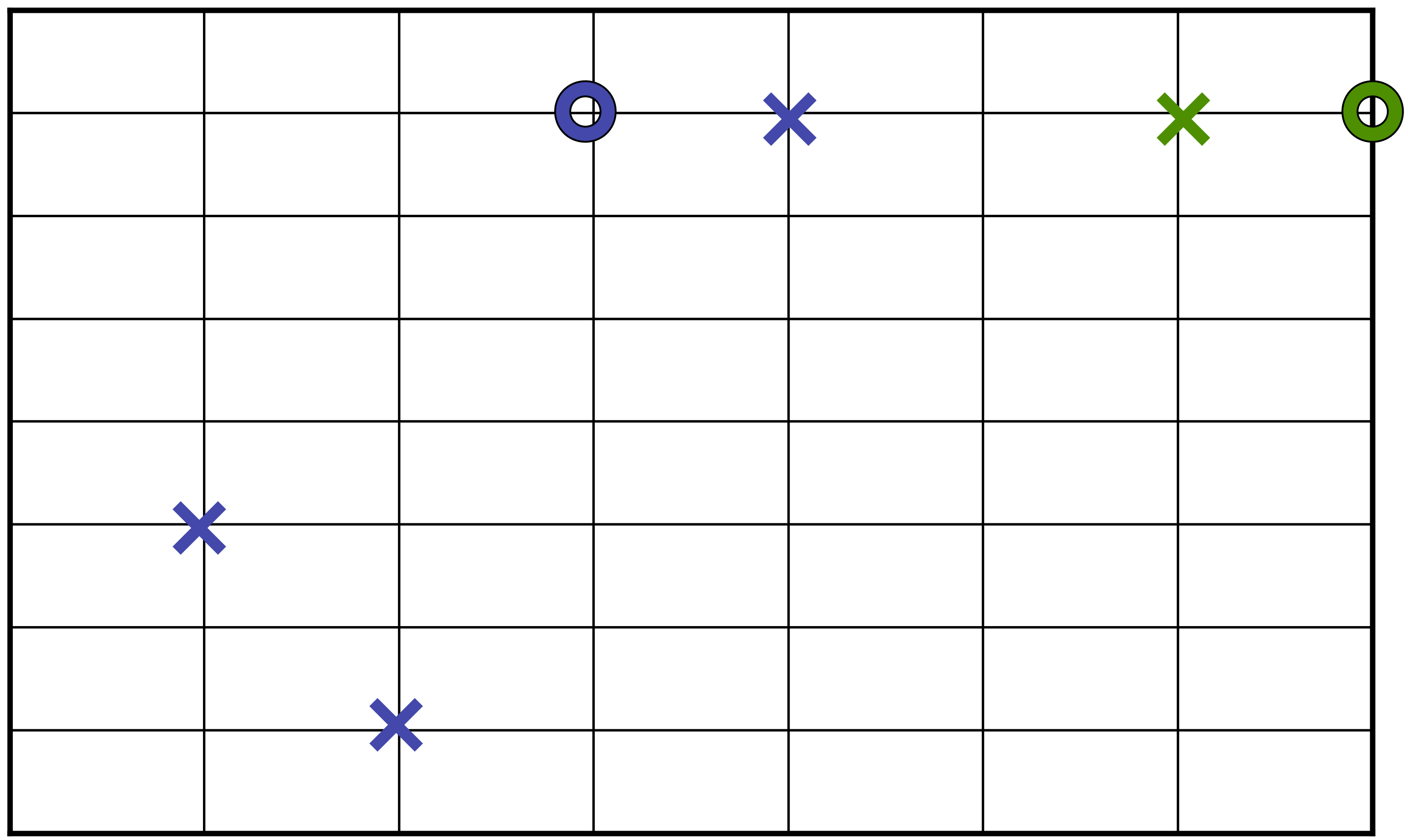
K-means

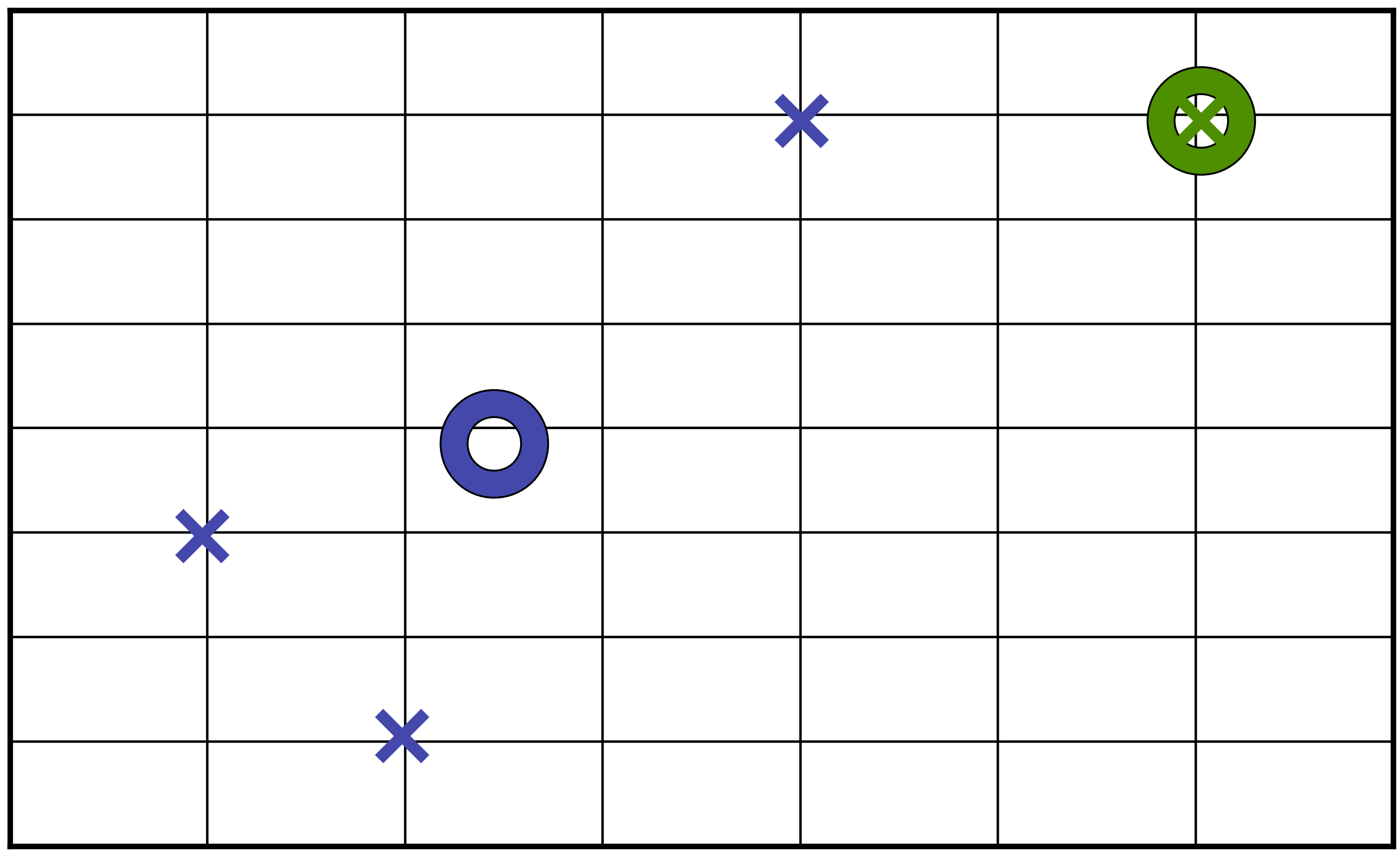
- Take n vectors and cluster into k classes
  - $x_1=(2,1)$ $x_2=(1,3)$ $x_3=(6,7)$ $x_4=(4,7)$
- Pick k initial centers
  - $f_1=(4,3)$ $f_2=(5,5)$
- Distance (L_2 or L_1 norm) $d(x_i,x_j) = \sqrt{\sum_{k=1}^{n}(x_{i_k}-x_{j_k})^2}$
- Mean of n vectors

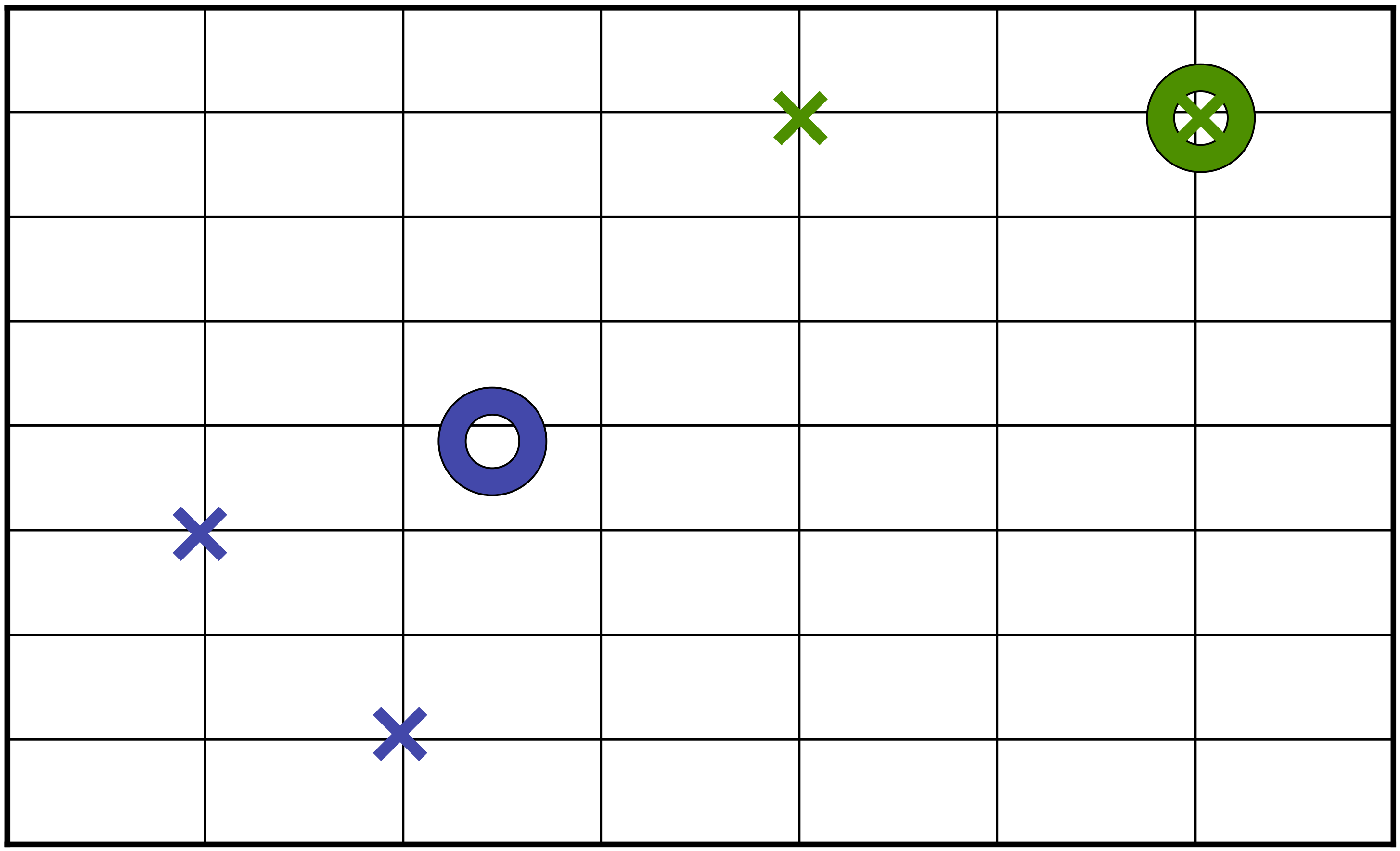
$$\mu(x_1,\ldots,x_n) = \left(\frac{\sum_{i=1}^{n} x_{i_1}}{n},\ldots,\frac{\sum_{i=1}^{n} x_{i_m}}{n}\right)$$

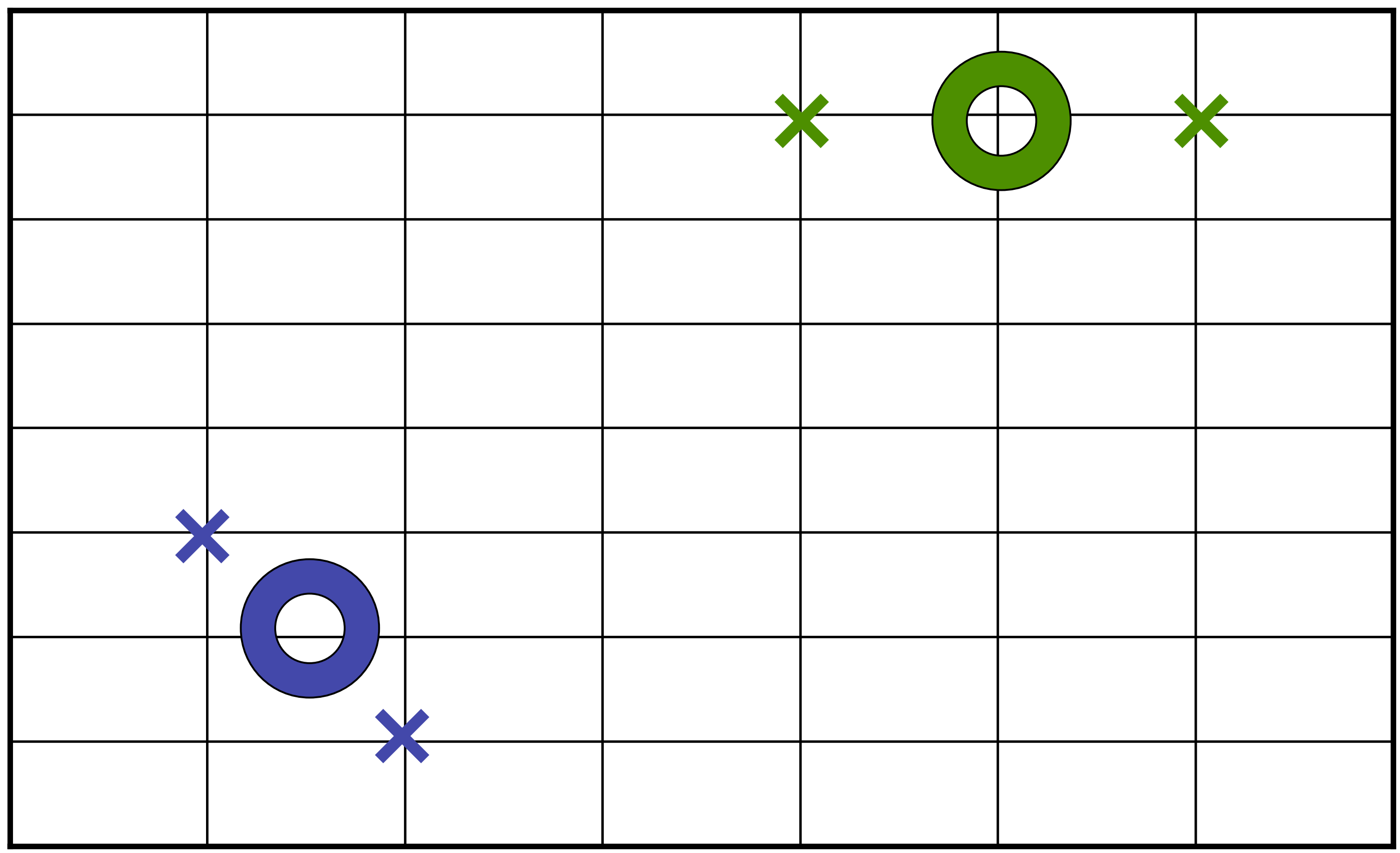












K-means algorithm

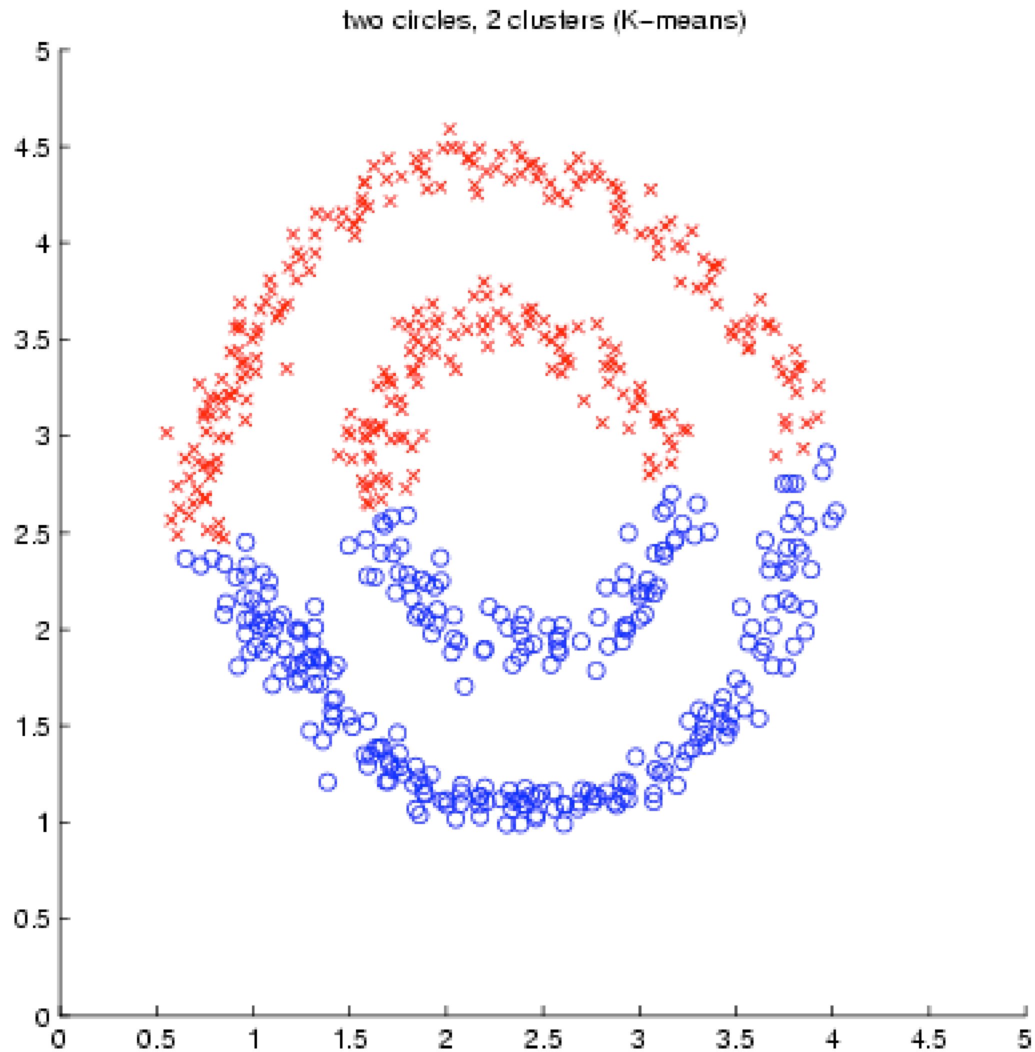
repeat until clusters do not change

for j=1..#clusters

$c_j = \{x_i : \text{for all } f_l \text{ and } d(x_i, f_j) <= d(x_i, f_l)\ \}$

for j=1..#clusters

$f_j = mean(c_j)$


two circles, 2 clusters (K−means)
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

EM algorithm

- Actually is a family of algorithms
- We've seen one example of an EM algorithm before: the forward-backward algorithm for HMMs
- EM can be used for any probability model where we can compute *sufficient statistics* E[likelihood] and max likelihood estimate

EM for Gaussian Mixtures

- Consider a set of points: x_1, …, x_n
- There are k clusters
- z is a 2d array where z_{ij} is 1 if x_i belongs to cluster j and 0 otherwise
- Each cluster j is defined as a Gaussian (normal) distribution

$$n(x;\mu_j,\sigma_j) = \frac{1}{\sqrt{2\pi}\sigma_j} exp\left[\frac{-(x-\mu_j)^2}{2\sigma_j^2}\right]$$

EM for Gaussian Mixtures

- We also multiply with a prior

$$\rho_j \cdot n(x;\mu_j,\sigma_j)$$

- So the probability of any item x_i is

$$p(x_i) = \sum_{j=1}^{k} \rho_j \cdot n(x_i,\mu_j,\sigma_j)$$

EM for Gaussian Mixtures

- Parameters

$$\theta_j = (\mu_j, \sigma_j, \rho_j)$$

$$\Theta = (\theta_1, \ldots, \theta_k)$$

- Log likelihood of the data *X* given the parameters

$$L(X \mid \Theta) = log \prod_i P(x_i) = log \prod_i \sum_j \rho_j \cdot n(x_i; \mu_j, \sigma_j)$$

$$\sum_i log \sum_j \rho_j \cdot n(x_i; \mu_j, \sigma_j)$$

EM for Gaussian Mixtures

- Find expected values of the hidden parameters z_{ij} and use that to compute Maximum Likelihood estimates

$$E(z_{ij} \mid x_i; \Theta) = \frac{\rho_j \cdot n(x_i; \mu_j, \sigma_j)}{p(x_i)}$$

$$\mu'_j = \frac{\sum_i E(z_{ij} \mid x_i) \cdot x_i}{\sum_i E(z_{ij} \mid x_i)} \qquad \sigma'_j = \frac{\sum_i E(z_{ij} \mid x_i) \cdot (x_i - \mu'_j)^2}{\sum_i E(z_{ij} \mid x_i)}$$

EM for Gaussian Mixtures

- We start with an initial setting for the parameters and iterate until convergence
- Convergence is guaranteed to a local optimum based on a theorem by Dempster, Laird and Rubin, 1977
- We only considered scalar x_i (for vector x_i we need multivariate Gaussians)
- K-means is a special case of using EM for Gaussian mixtures for clustering