Clustering

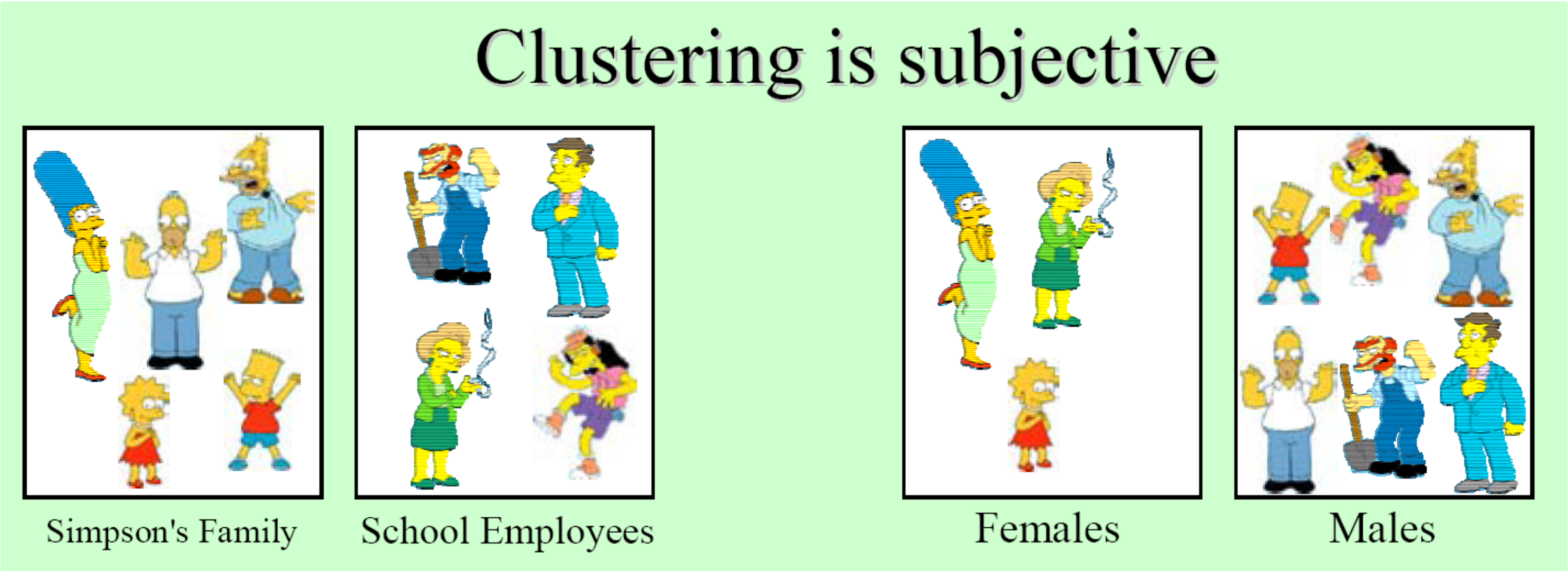
Aarti Singh

Machine Learning 10-315
Nov 11, 2019

Some slides courtesy of Eric Xing, Carlos Guestrin

What is clustering?

- Clustering: the process of grouping a set of objects into classes of similar objects
  - high intra-class similarity
  - low inter-class similarity
  - It is the most common form of unsupervised learning



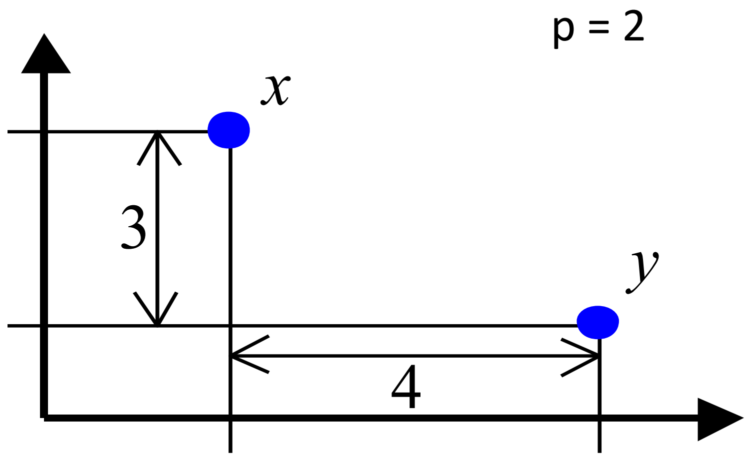
What is Similarity?

Hard to define! But *we know it when we see it*

- The real meaning of similarity is a philosophical question. We will take a more pragmatic approach - think in terms of a distance (rather than similarity) between vectors or correlations between random variables.

Distance metrics

$x = (x_1, x_2, \ldots, x_p)$
$y = (y_1, y_2, \ldots, y_p)$



Euclidean distance $$d(x,y) = \sqrt[2]{\sum_{i=1}^{p} |x_i - y_i|^2}$$ **5**

Manhattan distance $$d(x,y) = \sum_{i=1}^{p} |x_i - y_i|$$ **7**

Sup-distance $$d(x,y) = \max_{1 \le i \le p} |x_i - y_i|$$ **4**

Correlation coefficient

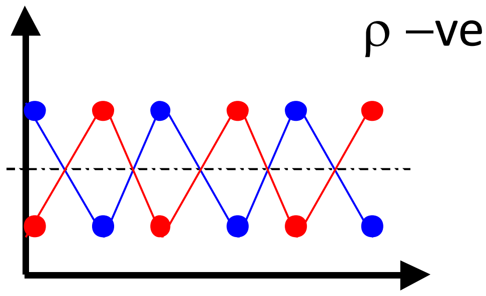
$x = (x_1, x_2, \ldots, x_p)$
$y = (y_1, y_2, \ldots, y_p)$

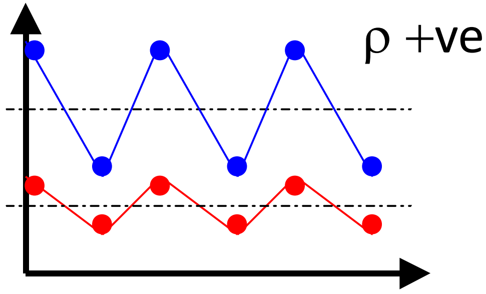
Random vectors (e.g. expression levels of two genes under various drugs)

Pearson correlation coefficient

$$\rho(x,y) = \frac{\sum_{i=1}^{p}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{p}(x_i - \bar{x})^2 \times \sum_{i=1}^{p}(y_i - \bar{y})^2}}$$

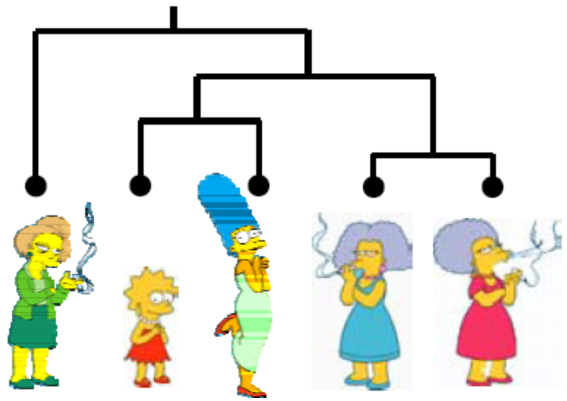
$$\text{where } \bar{x} = \frac{1}{p}\sum_{i=1}^{p} x_i \text{ and } \bar{y} = \frac{1}{p}\sum_{i=1}^{p} y_i .$$



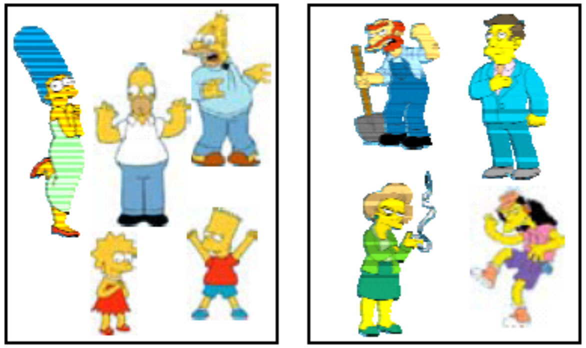


Clustering Algorithms

- Hierarchical algorithms
  - Single-linkage
  - Average-linkage
  - Complete-linkage
  - Centroid-based



- Partition algorithms
  - K means clustering
  - Mixture-Model based clustering



Hierarchical Clustering

- Bottom-Up Agglomerative Clustering

  Starts with each object in a separate cluster, and repeat:
  - Joins the most similar pair of clusters,
  - Update the similarity of the new cluster to others

  until there is only one cluster.

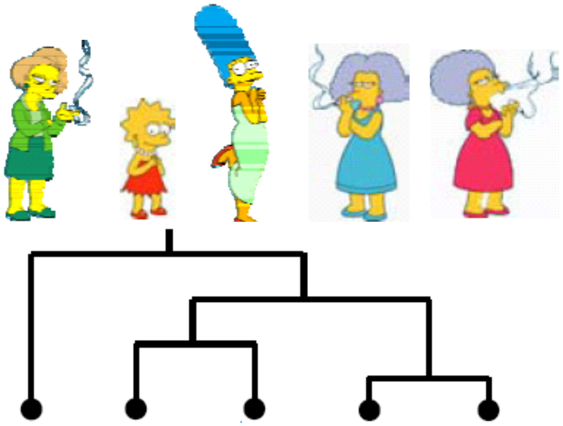
  Greedy - less accurate but simple to implement

- Top-Down divisive

  Starts with all the data in a single cluster, and repeat:
  - Split each cluster into two using a partition algorithm

  Until each object is a separate cluster.

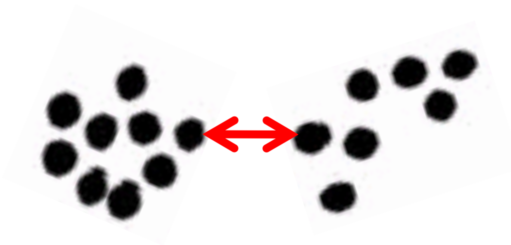
  More accurate but complex to implement



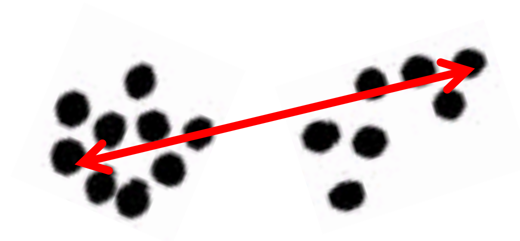
Bottom-up Agglomerative clustering

Different algorithms differ in how the similarities are defined (and hence updated) between two clusters

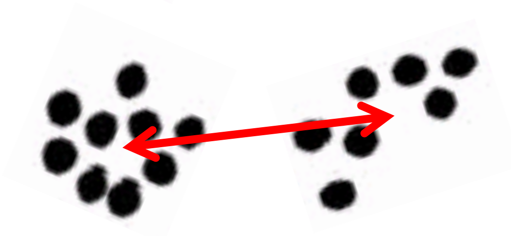
- Single-Linkage
  - Nearest Neighbor: similarity between their closest members.



- Complete-Linkage
  - Furthest Neighbor: similarity between their furthest members.



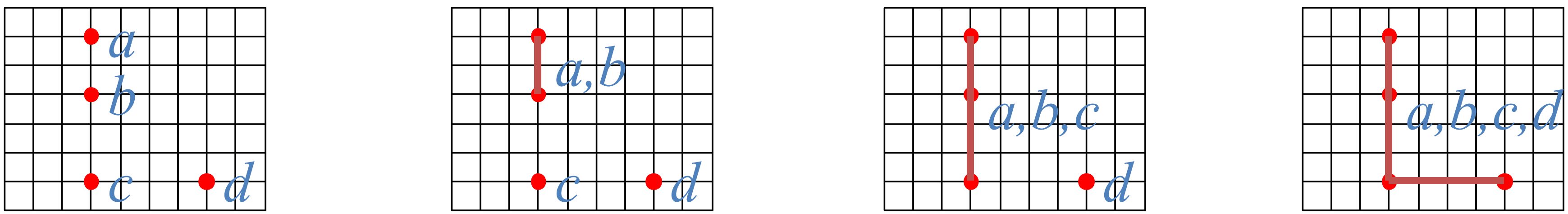
- Centroid
  - Similarity between the centers of gravity



- Average-Linkage
  - Average similarity of all cross-cluster pairs.

Single-Linkage Method

Euclidean Distance



(1) (2) (3)

	b	c	d
a	2	5	6
b		3	5
c			4

	b	c	d
a	2	5	6
b		3	5
c			4

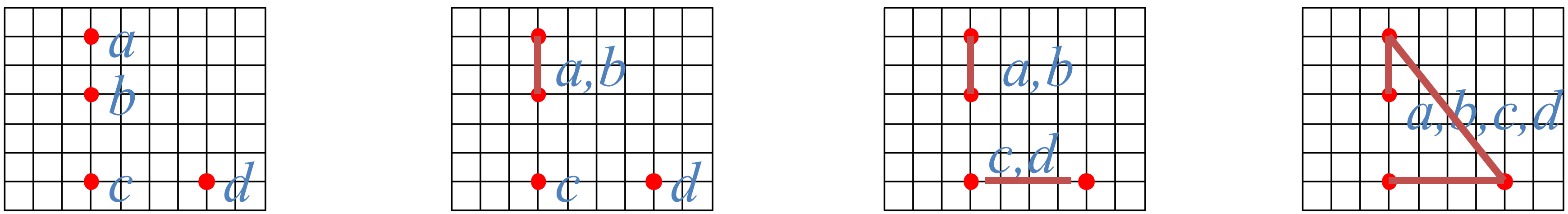
	c	d
a,b	3	5
c		4

	d
a,b,c	4

Distance Matrix

Complete-Linkage Method

Euclidean Distance



(1) (2) (3)

	b	*c*	*d*
a	2	5	6
b		3	5
c			4

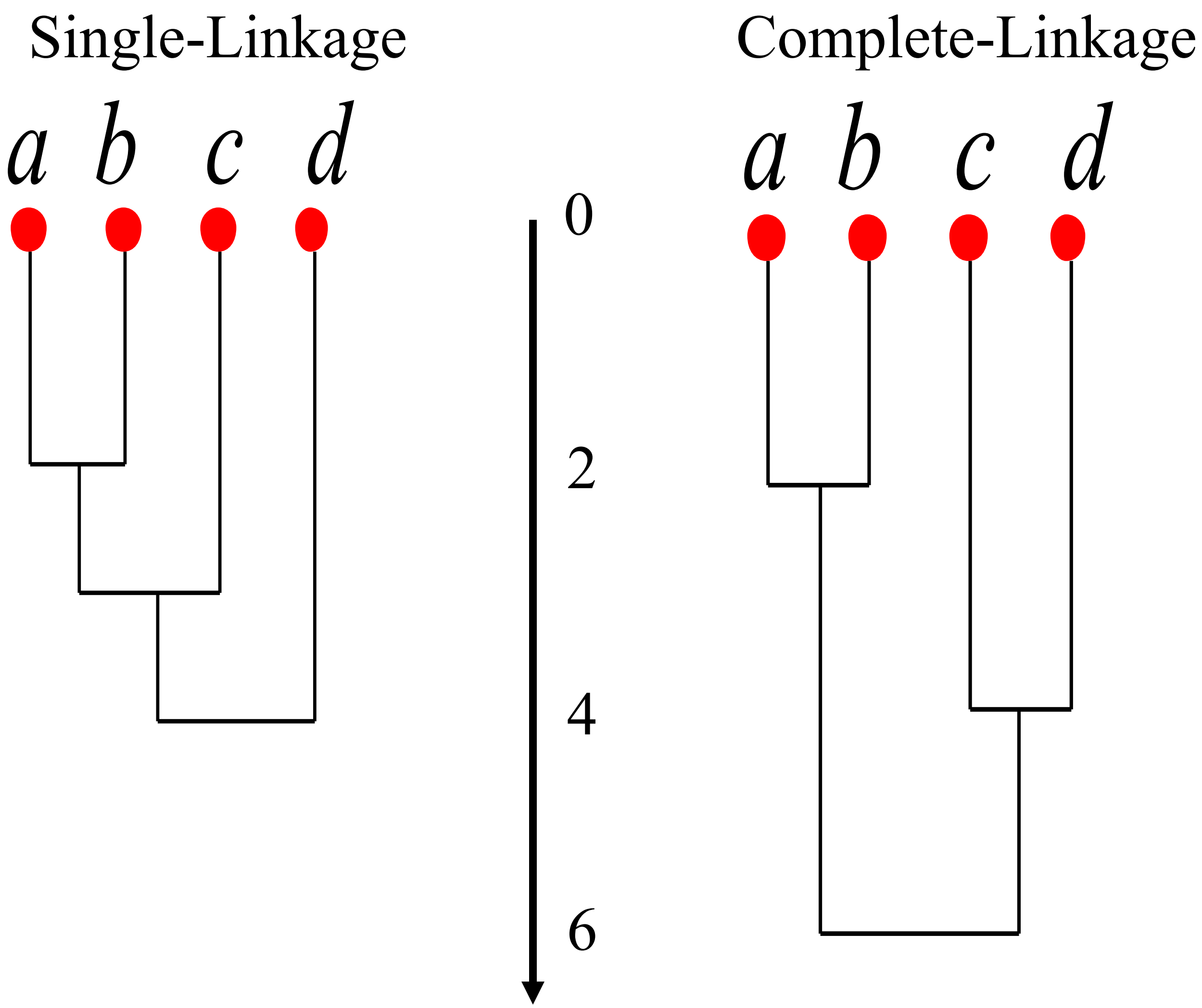
	b	*c*	*d*
a	2	5	6
b		3	5
c			4

	c	*d*
a,b	5	6
c		4

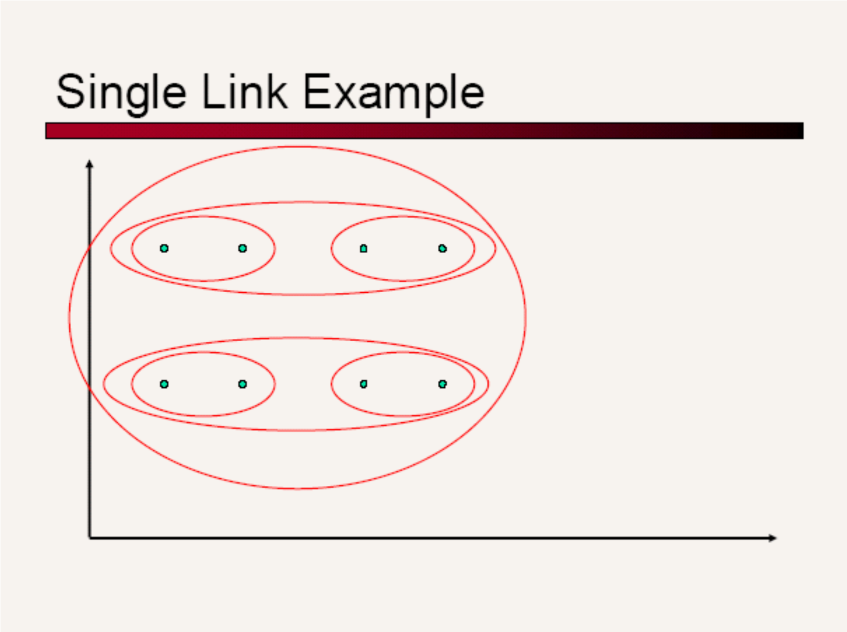
	c,d
a,b	6

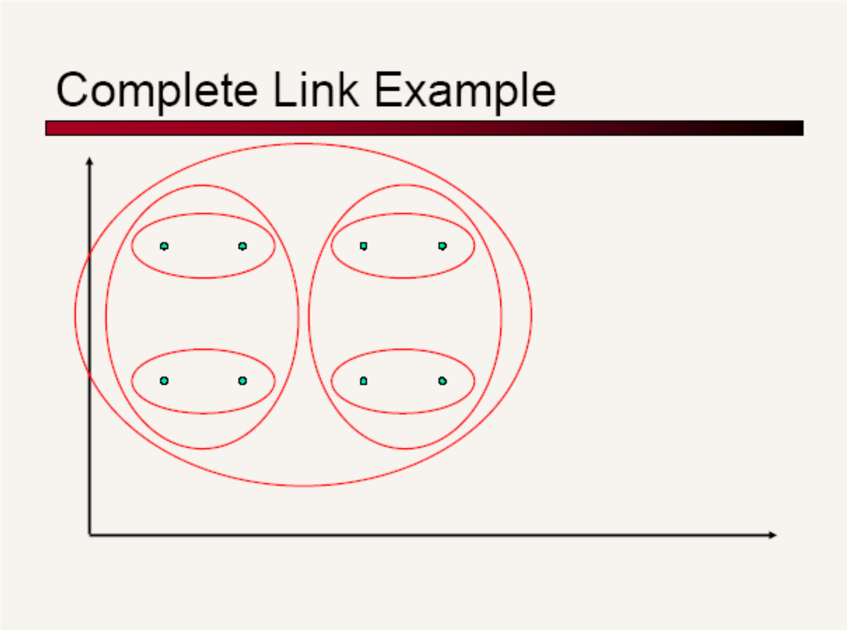
Distance Matrix

Dendrograms



Another Example





Single vs. Complete Linkage

Shape of clusters

Single-linkage	allows anisotropic and non-convex shapes
Complete-linkage	assumes isotopic, convex shapes

Computational Complexity

- All hierarchical clustering methods need to compute similarity of all pairs of *n* individual instances which is $O(n^2)$.

- At each iteration,
  - Sort similarities to find largest one $O(n^2 \log n)$.
  - Update similarity between merged cluster and other clusters.

    Computing similarity to each other cluster can be done in constant time.

- So we get $O(n^2 \log n)$ or $O(n^3)$ (if naïvely implemented)

Partitioning Algorithms

- Partitioning method: Construct a partition of *n* objects into a set of *K* clusters
- Given: a set of objects and the number *K*
- Find: a partition of *K* clusters that optimizes the chosen partitioning criterion
  - Globally optimal: exhaustively enumerate all partitions
  - Effective heuristic method: K-means algorithm

K-Means

Algorithm

Input – Desired number of clusters, *k*

Initialize – the *k* cluster centers (randomly if necessary)

Iterate –

1. Assign points to the nearest cluster centers

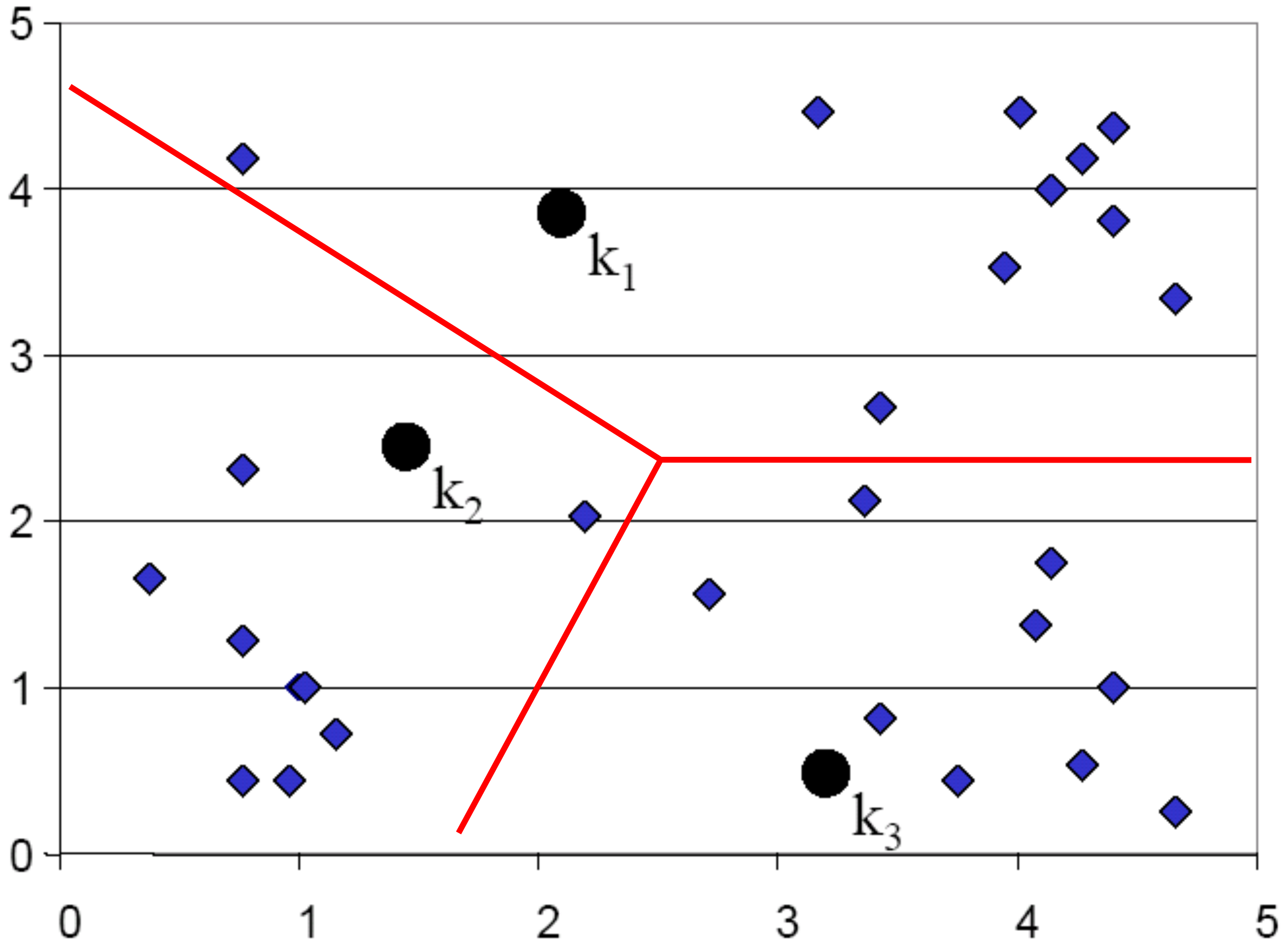
2. Re-estimate the *k* cluster centers (aka the centroid or mean), by assuming the memberships found above are correct.

$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

Termination –

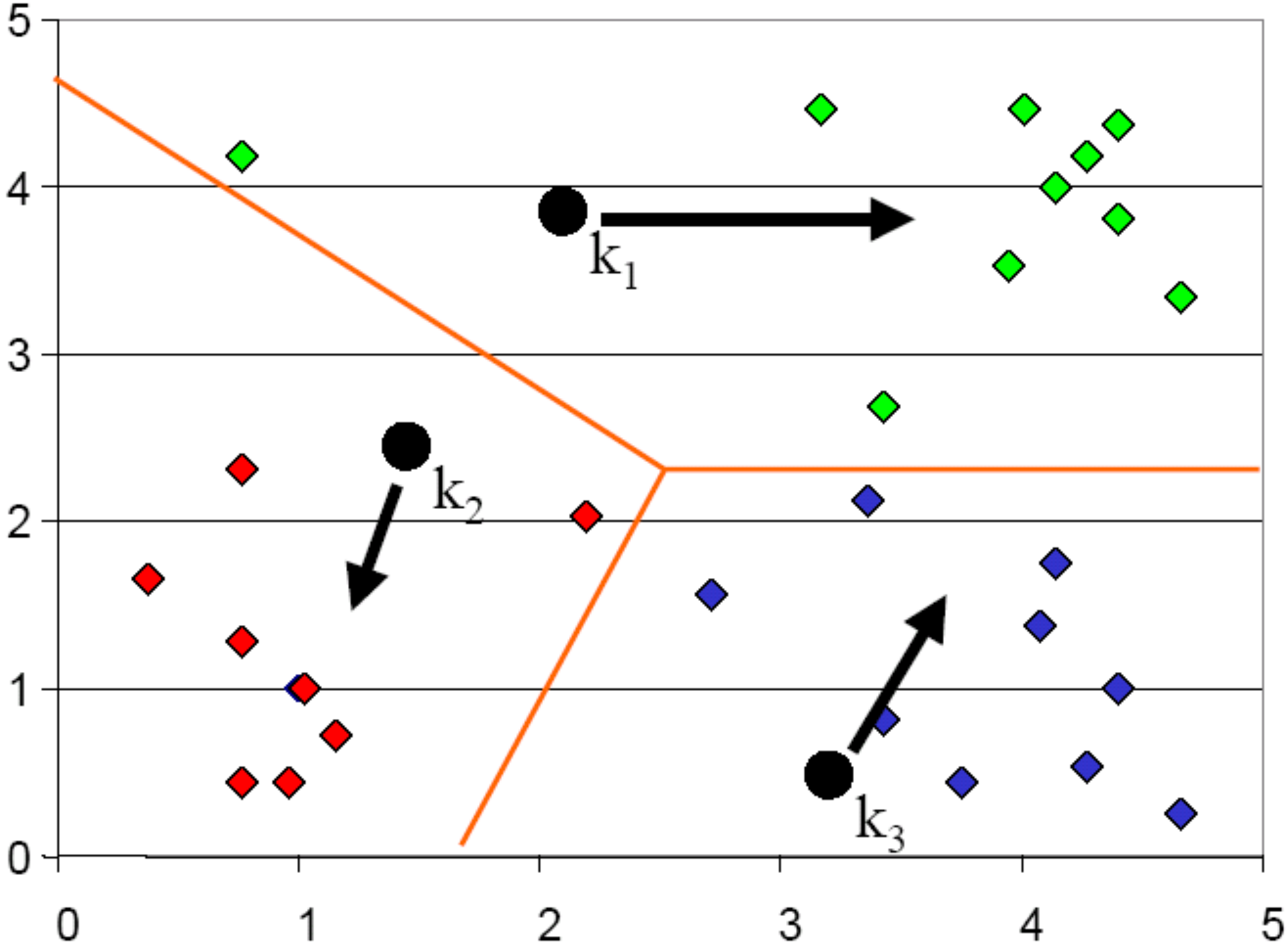
If none of the objects changed membership in the last iteration, exit. Otherwise go to 1.

K-means Clustering: Step 1

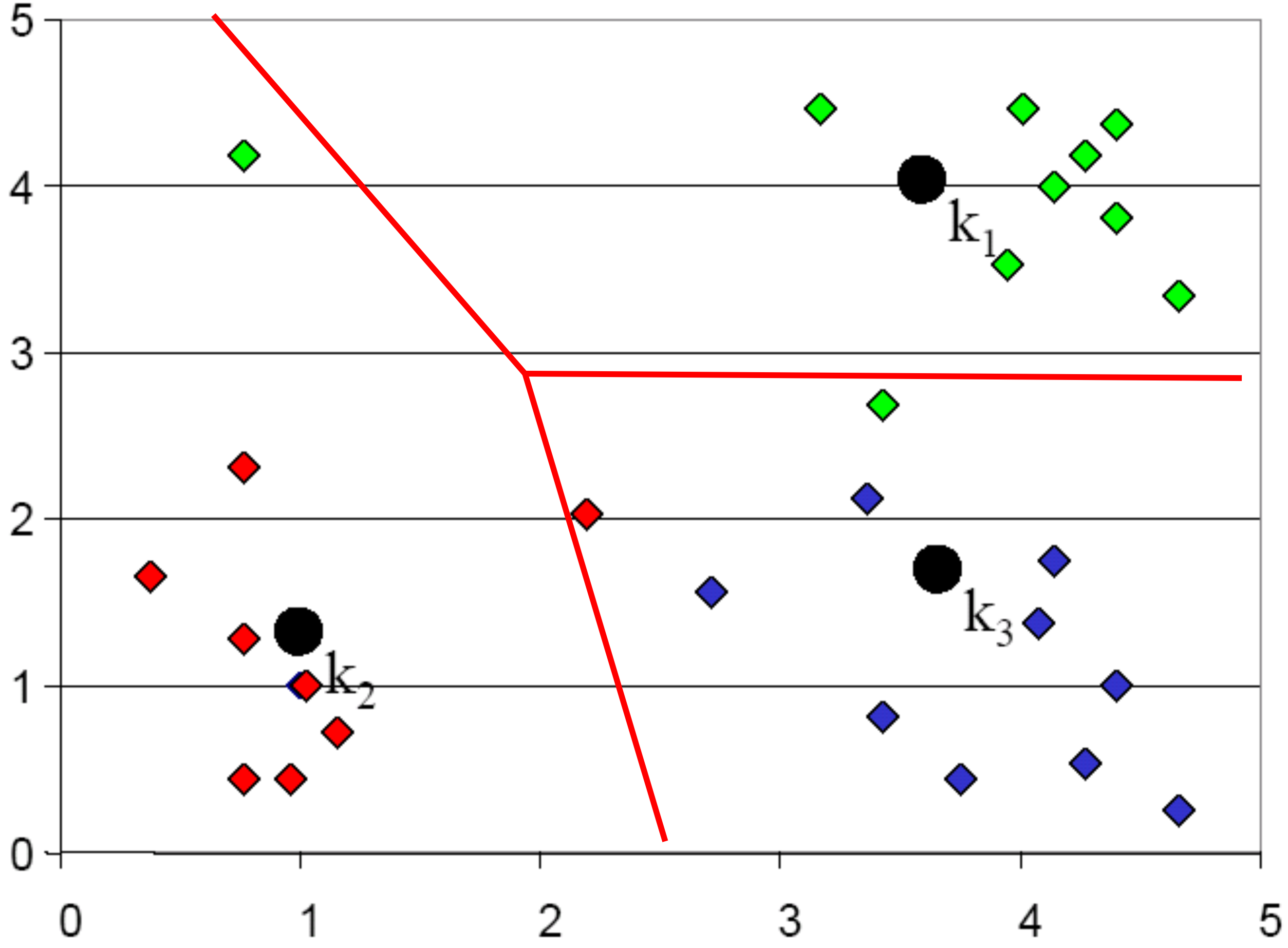


Voronoi diagram

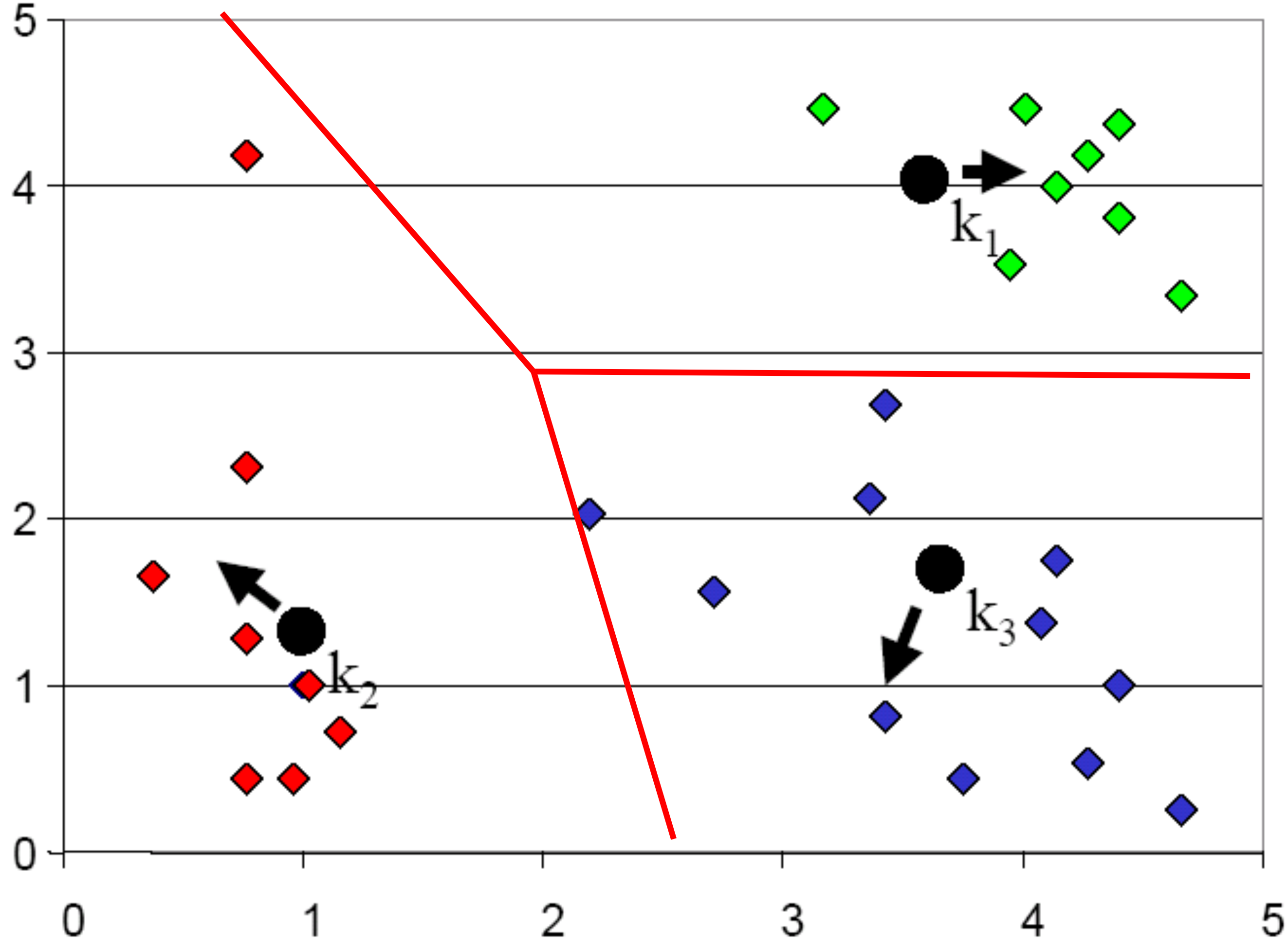
K-means Clustering: Step 2



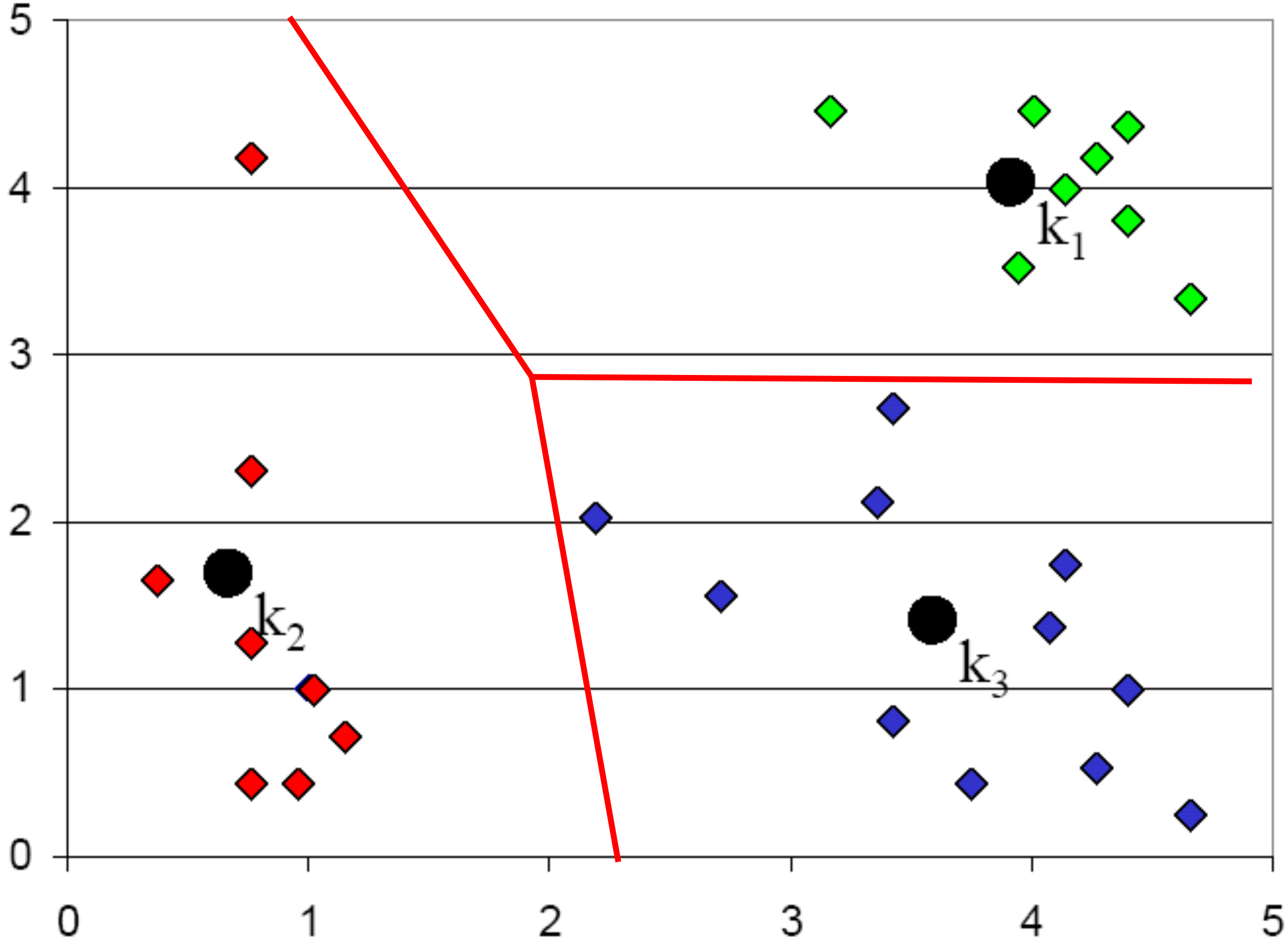
K-means Clustering: Step 3



K-means Clustering: Step 4



K-means Clustering: Step 5



K-means Recap …

- Randomly initialize *k* centers
  - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$

K-means Recap …

- Randomly initialize *k* centers
  - $\mu^{(0)} = \mu_1^{(0)},\ldots,\mu_k^{(0)}$

Iterate t = 0, 1, 2, …

- **Classify**: Assign each point $j \in \{1,\ldots m\}$ to nearest center:
  - $C^{(t)}(j) \leftarrow \arg \min_{i=1,\ldots,k} \|\mu_i^{(t)} - x_j\|^2$

K-means Recap …

- Randomly initialize *k* centers
  - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$

Iterate t = 0, 1, 2, …

- **Classify**: Assign each point $j \in \{1, \ldots m\}$ to nearest center:
  - $C^{(t)}(j) \leftarrow \arg\min_{i=1,\ldots,k} \|\mu_i^{(t)} - x_j\|^2$
- **Recenter**: μ_i becomes centroid of its points:
  - $\mu_i^{(t+1)} \leftarrow \arg\min_{\mu} \sum_{j:C^{(t)}(j)=i} \|\mu - x_j\|^2 \qquad i \in \{1, \ldots, k\}$
  - Equivalent to $\mu_i \leftarrow$ average of its points!

What is K-means optimizing?

- Potential function F(**μ**,C) of centers **μ** and point allocations C:

$$F(\mu, C) = \sum_{j=1}^{m} ||\mu_{C(j)} - x_j||^2$$

$$= \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Optimal K-means:
  - $\min_{\mu} \min_{C} F(\mu, C)$

K-means algorithm

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- **K-means algorithm:** (coordinate descent on F)

(1) Fix μ, optimize C — **Expected** cluster assignment

(2) Fix C, optimize μ — **Maximum** likelihood for center

Next class, we will see a generalization of this approach:

EM algorithm

Computational Complexity

- At each iteration,
  - Computing distance between each of the n objects and the K cluster centers is O(Kn).
  - Computing cluster centers: Each object gets added once to some cluster: O(n).

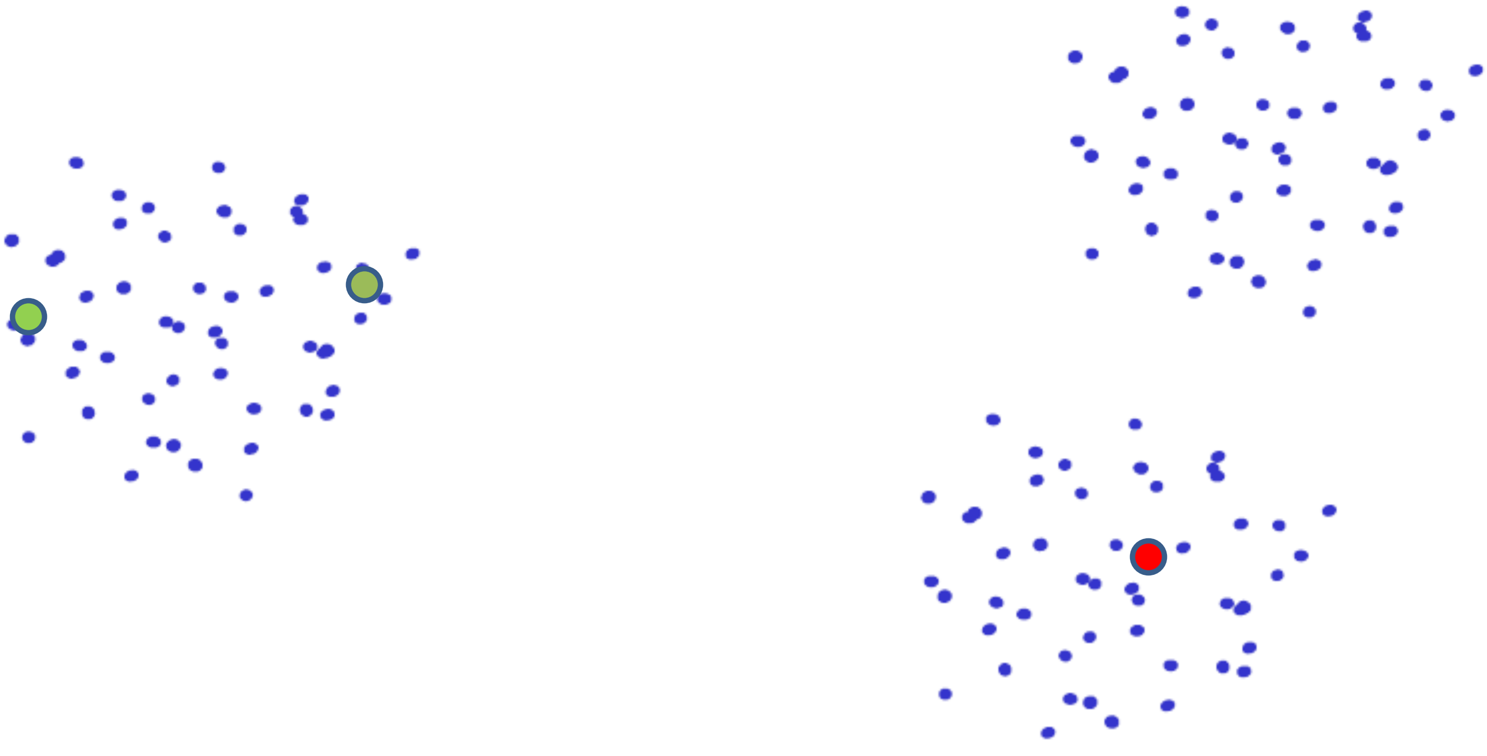
- Assume these two steps are each done once for l iterations: O(lKn).

Seed Choice

- Results are quite sensitive to seed selection.

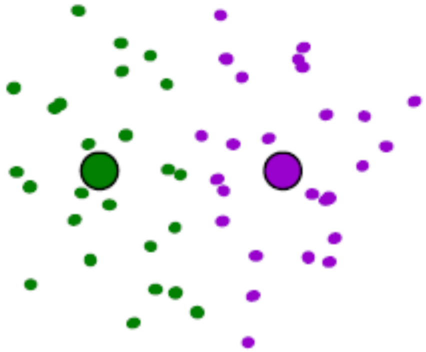
Seed Choice

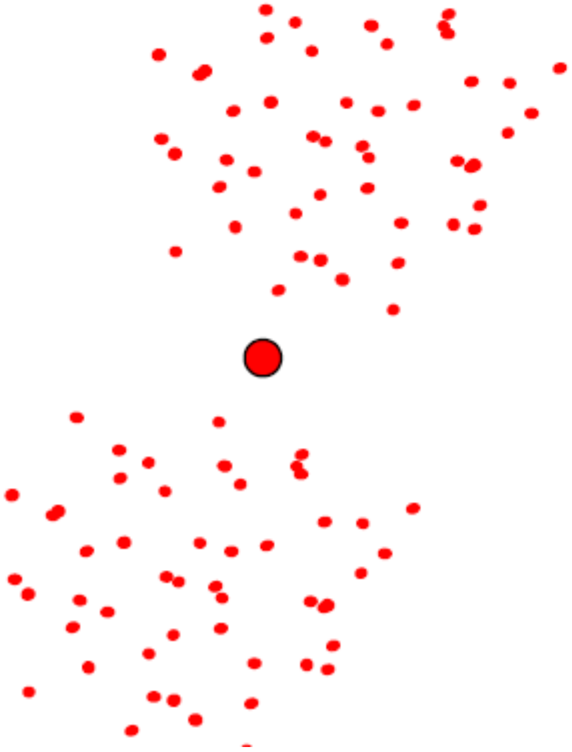
- Results are quite sensitive to seed selection.



Seed Choice

- Results are quite sensitive to seed selection.





Seed Choice

- Results can vary based on random seed selection.

- Some seeds can result in poor convergence rate, or convergence to sub-optimal clustering.
  - Try out multiple starting points (very important!!!)
  - k-means ++ algorithm of Arthur and Vassilvitskii

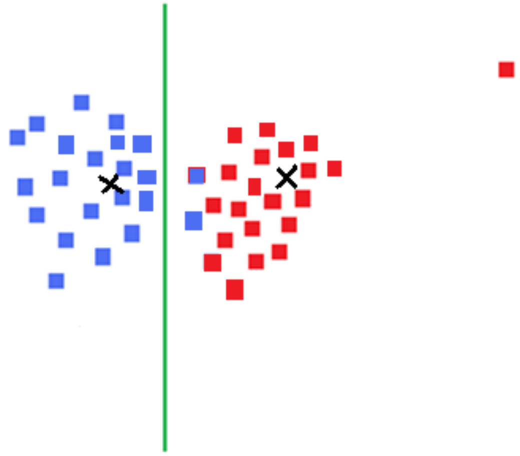
    key idea: choose centers that are far apart

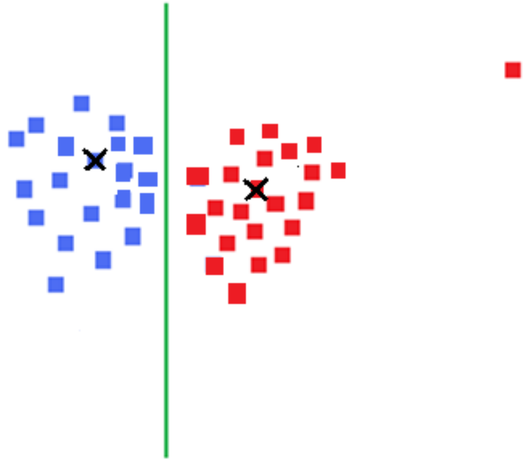
    (probability of picking a point as cluster center $\propto$ distance from nearest center picked so far)

Other Issues

- Shape of clusters
  - Assumes isotropic, equal variance, convex clusters

- Sensitive to Outliers
  - use K-medoids



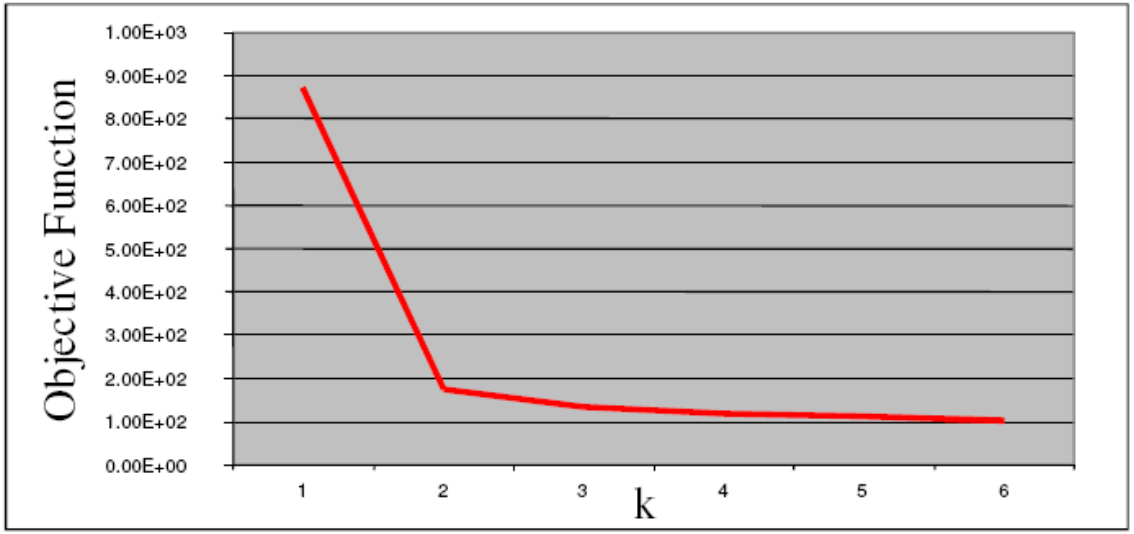


Other Issues

- Number of clusters K
  - Objective function

$$\sum_{j=1}^{m} ||\mu_{C(j)} - x_j||^2$$

  - Look for “Knee” in objective function



  - Can you pick K by minimizing the objective over K?