

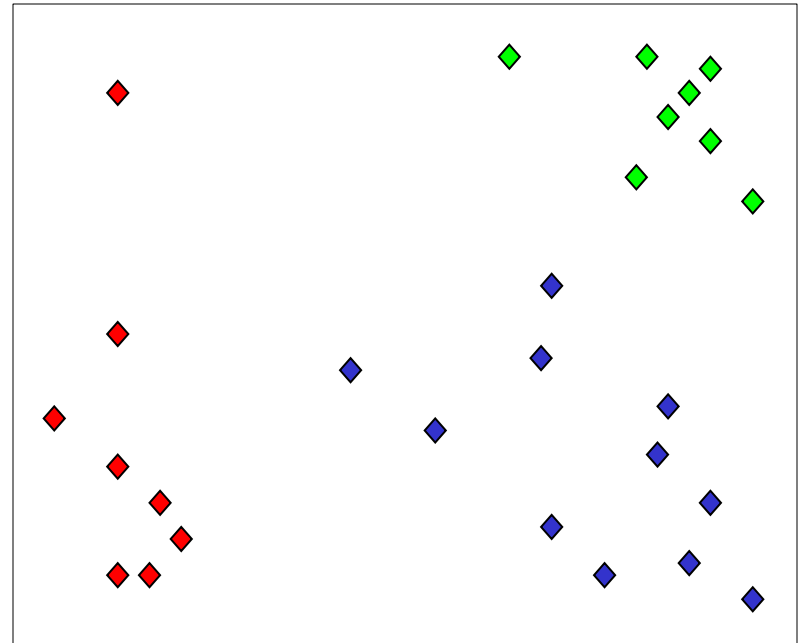
10601

Machine Learning

Clustering

What is Clustering?

- Organizing data into *clusters* such that there is
 - high intra-cluster similarity
 - low inter-cluster similarity
- Informally, finding natural groupings among objects.
- Why do we want to do that?
- Any REAL application?

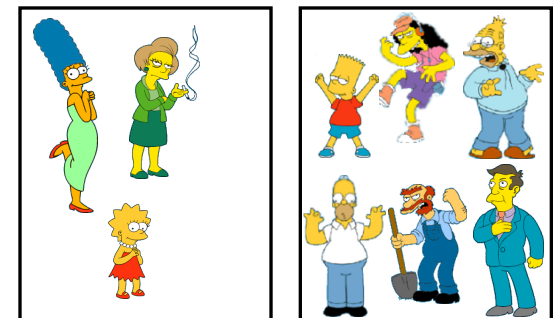
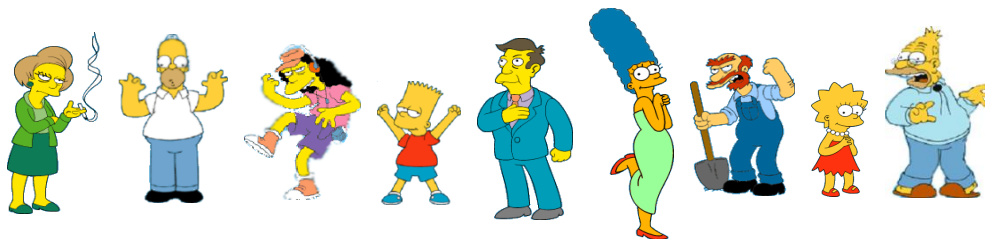


Outline

- Motivation
- Distance functions
- Hierarchical clustering
- Partitional clustering
 - K-means
 - Gaussian Mixture Models
- Number of clusters

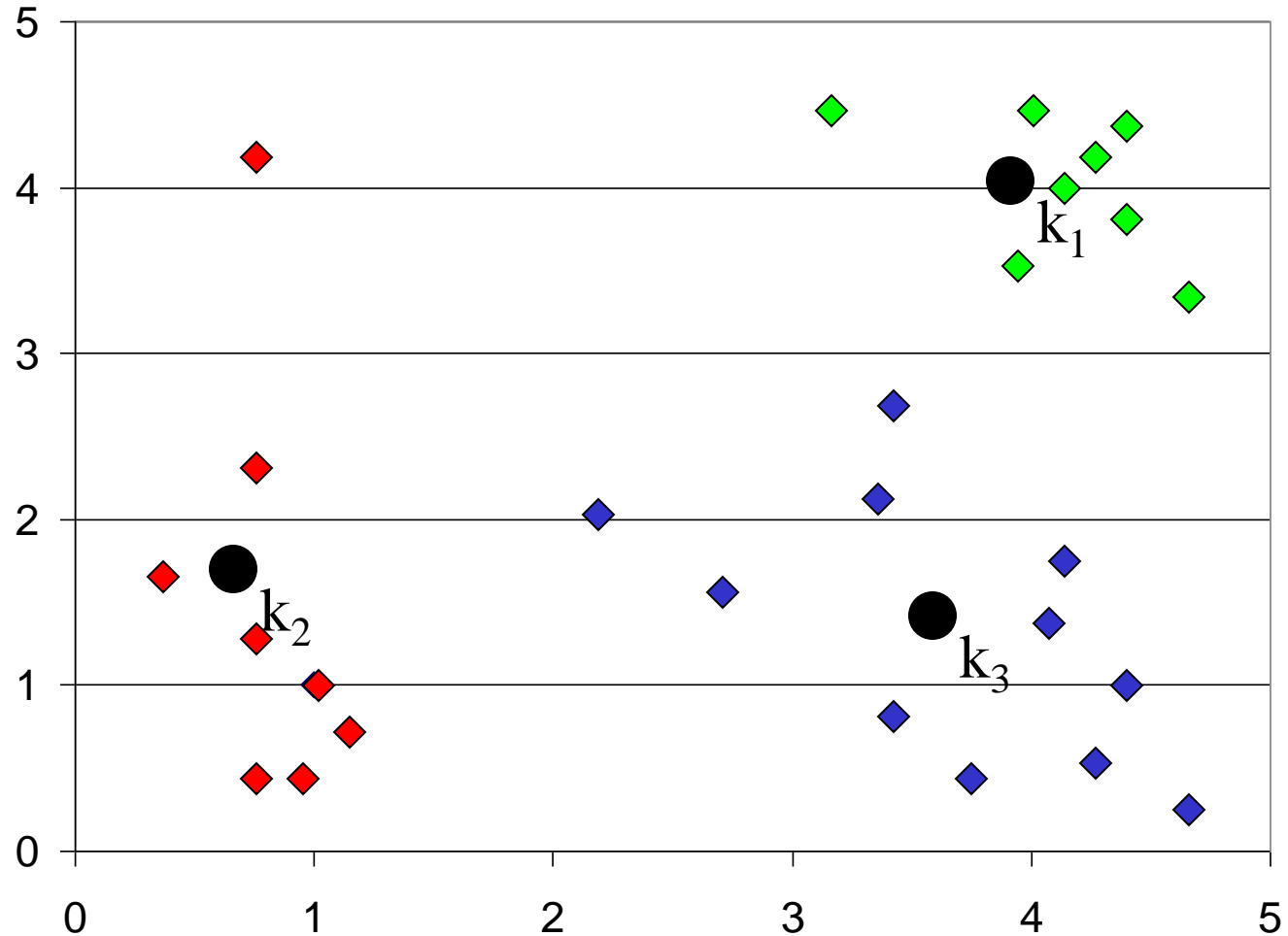
Partitional Clustering

- Nonhierarchical, each instance is placed in exactly one of K non-overlapping clusters.
- Since the output is only one set of clusters the user has to specify the desired number of clusters K .



K-means Clustering: Finished!

Re-assign and move centers, until ...
no objects changed membership.



Algorithm *k-means*

1. Decide on a value for K , the number of clusters.
2. Initialize the K cluster centers (randomly, if necessary).
3. Decide the class memberships of the N objects by assigning them to the nearest cluster center.
4. Re-estimate the K cluster centers, by assuming the memberships found above are correct.
5. Repeat 3 and 4 until none of the N objects changed membership in the last iteration.

Algorithm *k-means*

1. Decide on a value for K , the number of clusters (if necessary).
2. Initialize the K cluster centers (e.g., randomly or by hand).
Use one of the distance / similarity functions we discussed earlier
3. Decide the class memberships of the N objects by assigning them to the nearest cluster center.
4. Re-estimate the K cluster centers, by assuming the memberships found above are correct.
Average / median of class members
5. Repeat 3 and 4 until none of the N objects changed membership in the last iteration

Summary: *K-Means*

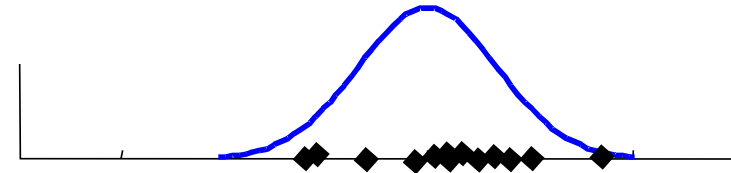
- Strength
 - Simple, easy to implement and debug
 - Intuitive objective function: optimizes intra-cluster similarity
 - *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.
- Weakness
 - Applicable only when *mean* is defined, what about categorical data?
 - Often terminates at a *local optimum*. Initialization is important.
 - Need to specify K , the *number* of clusters, in advance
 - Unable to handle noisy data and *outliers*
 - Not suitable to discover clusters with *non-convex shapes*
- Summary
 - Assign members based on current centers
 - Re-estimate centers based on current assignment

Outline

- Motivation
- Distance measure
- Hierarchical clustering
- Partitional clustering
 - K-means
 - Gaussian Mixture Models
 - Number of clusters

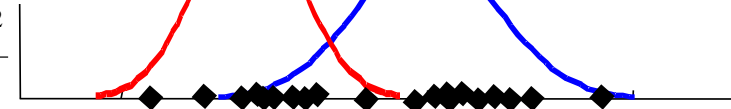
Gaussian Mixture Models

- Gaussian $P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\eta)^2}{2\sigma^2}}$
 - ex. height of one population



- Gaussian Mixture: Generative modeling framework

$$P(C=i) = w_i, \quad P(x|C=i) = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{(x-\eta_i)^2}{2\sigma_i^2}}$$



$$P(x) = \sum_i P(C=i, x) = \sum_i P(x|C=i)P(C=i) = \sum_i w_i \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{(x-\eta_i)^2}{2\sigma_i^2}}$$

– ex. height in two populations

Likelihood of a data point given the model

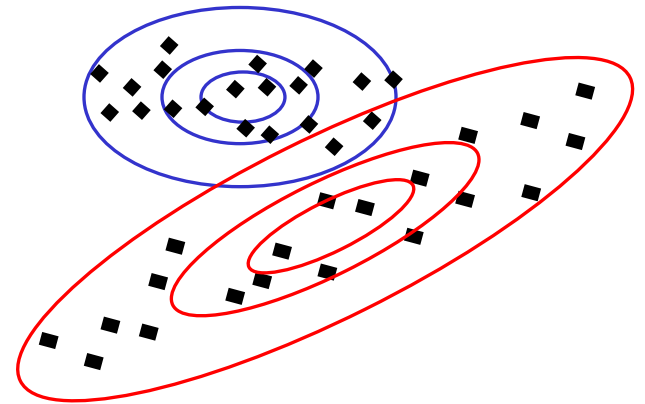
Gaussian Mixture Models

- Mixture of Multivariate Gaussian

$$P(C = i) = w_i$$

$$P(x | C = i) = \frac{1}{2\pi^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

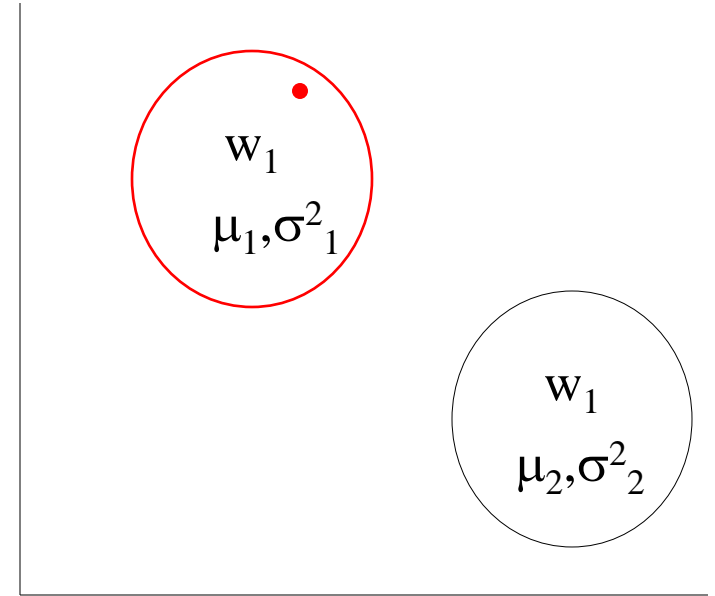
- ex. y-axis is blood pressure
and x-axis is age



GMM: A generative model

$$\sum_i w_i = 1$$

- Assuming we know the number of components (k), their weights (w_i) and parameters (μ_i, Σ_i) we can generate new instances from a GMM in the following way:
 - Pick one component at random with probability w_i for each component
 - Sample a point x from $N(\mu_i, \Sigma_i)$



Estimating model parameters

- We have a weight, mean and covariance parameters for each class
- As usual we can write the likelihood function for our model


$$p(x_1 \cdots x_n \mid \theta) = \prod_{j=1}^n \left(\sum_{i=1}^k p(x_j \mid C = i) w_i \right)$$

GMM+EM = “Soft K-means”

- Decide the number of clusters, K
- Initialize parameters (randomly)
- E-step: assign *probabilistic* membership to all input samples j

One for each
cluster

$$p_{i,j} = p(C=i | x_j) = \frac{p(x_j | C=i)p(C=i)}{\sum_k p(x_j | C=k)p(C=k)}$$


$$p_i = \sum_j p_{i,j}$$

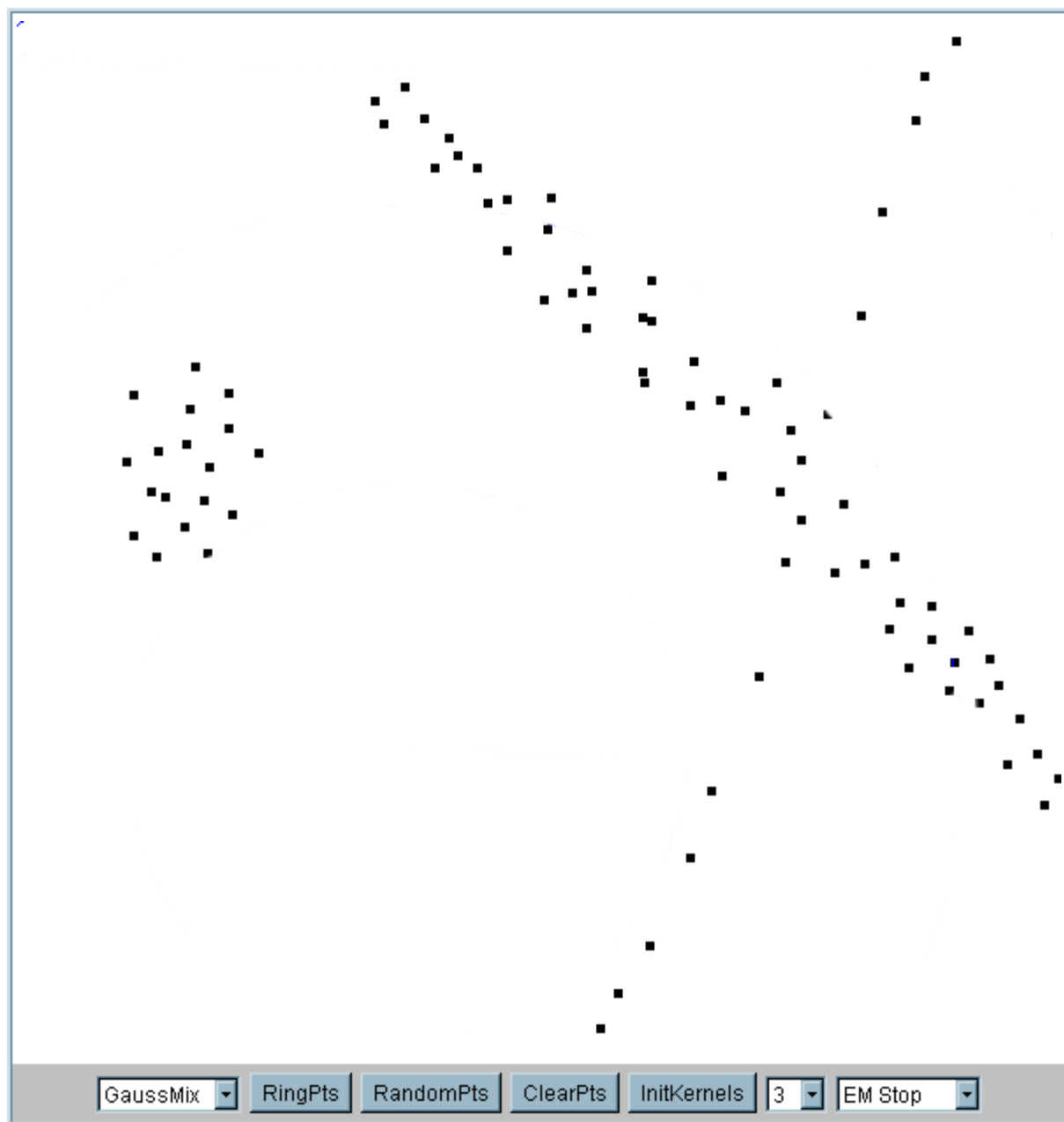
- M-step: re-estimate parameters based on *probabilistic* membership

$$\mu_i \leftarrow \sum_j \frac{p_{i,j} \mathbf{x}_j}{p_i}$$

$$\Sigma_i \leftarrow \sum_j \frac{p_{i,j} \mathbf{x}_j \mathbf{x}_j^T}{p_i}$$

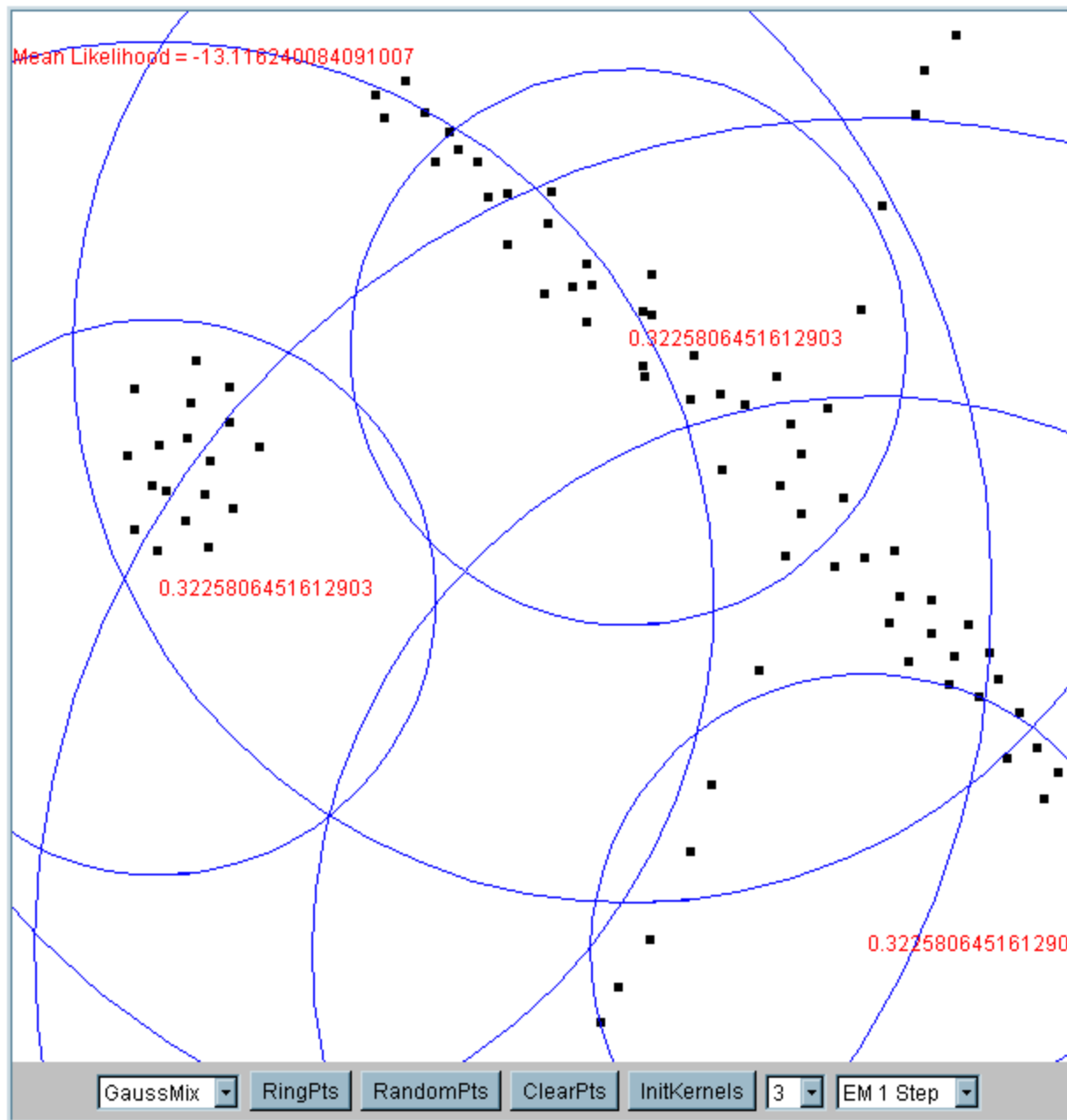
$$w_i = \frac{p_i}{\sum_j p_j}$$

- Repeat until change in parameters are smaller than a threshold

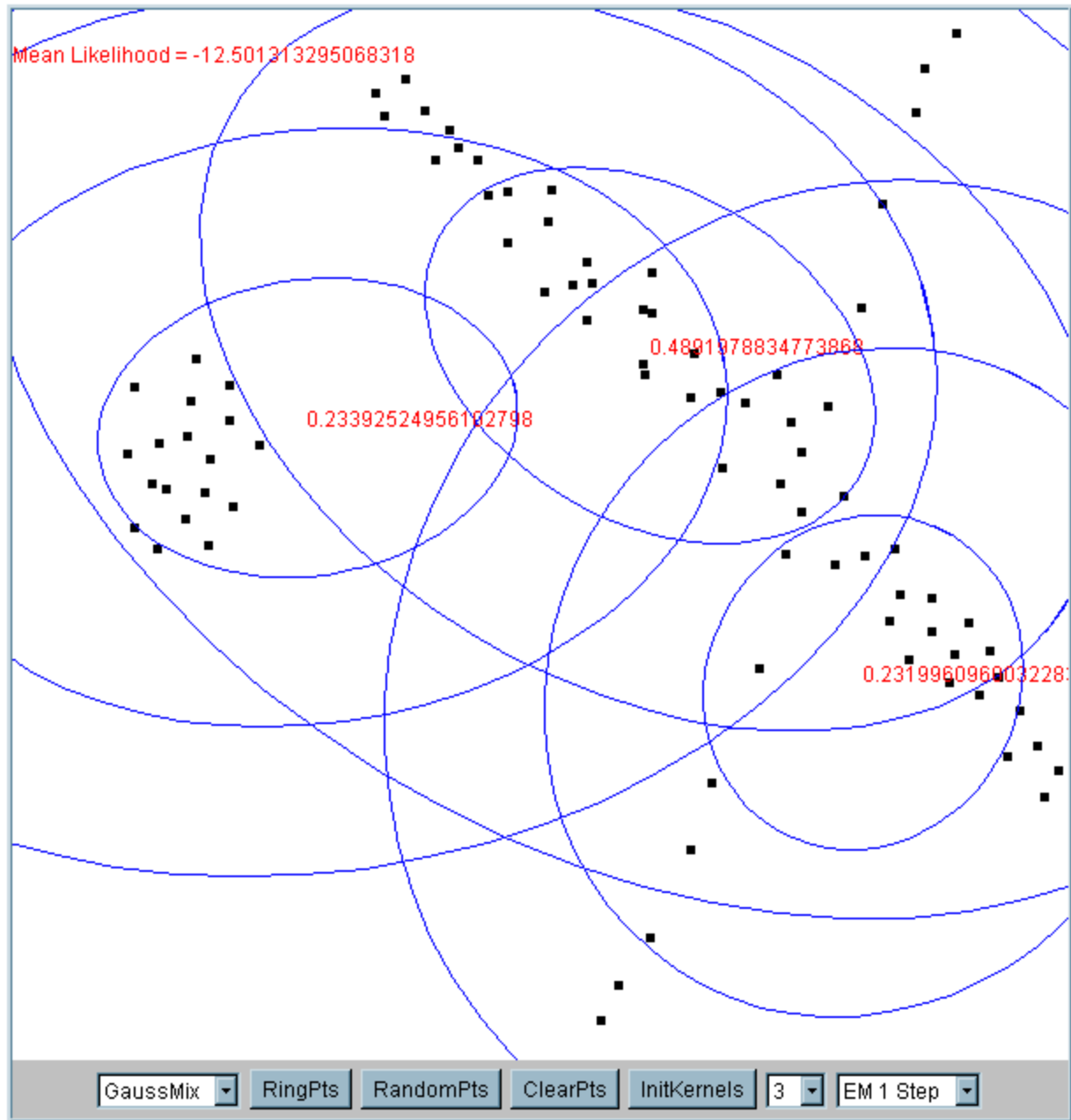


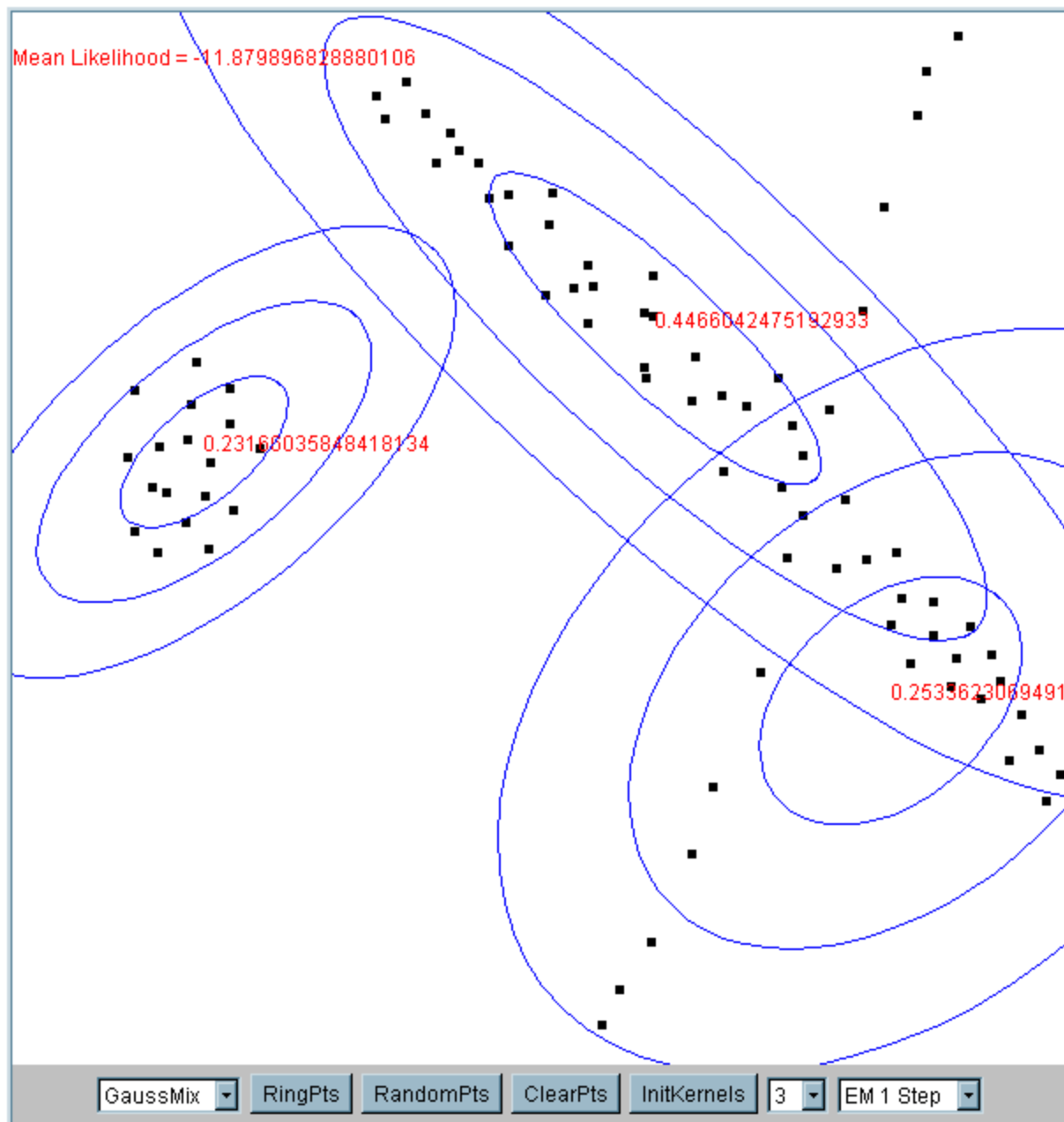
Iteration 1

The cluster means are randomly assigned



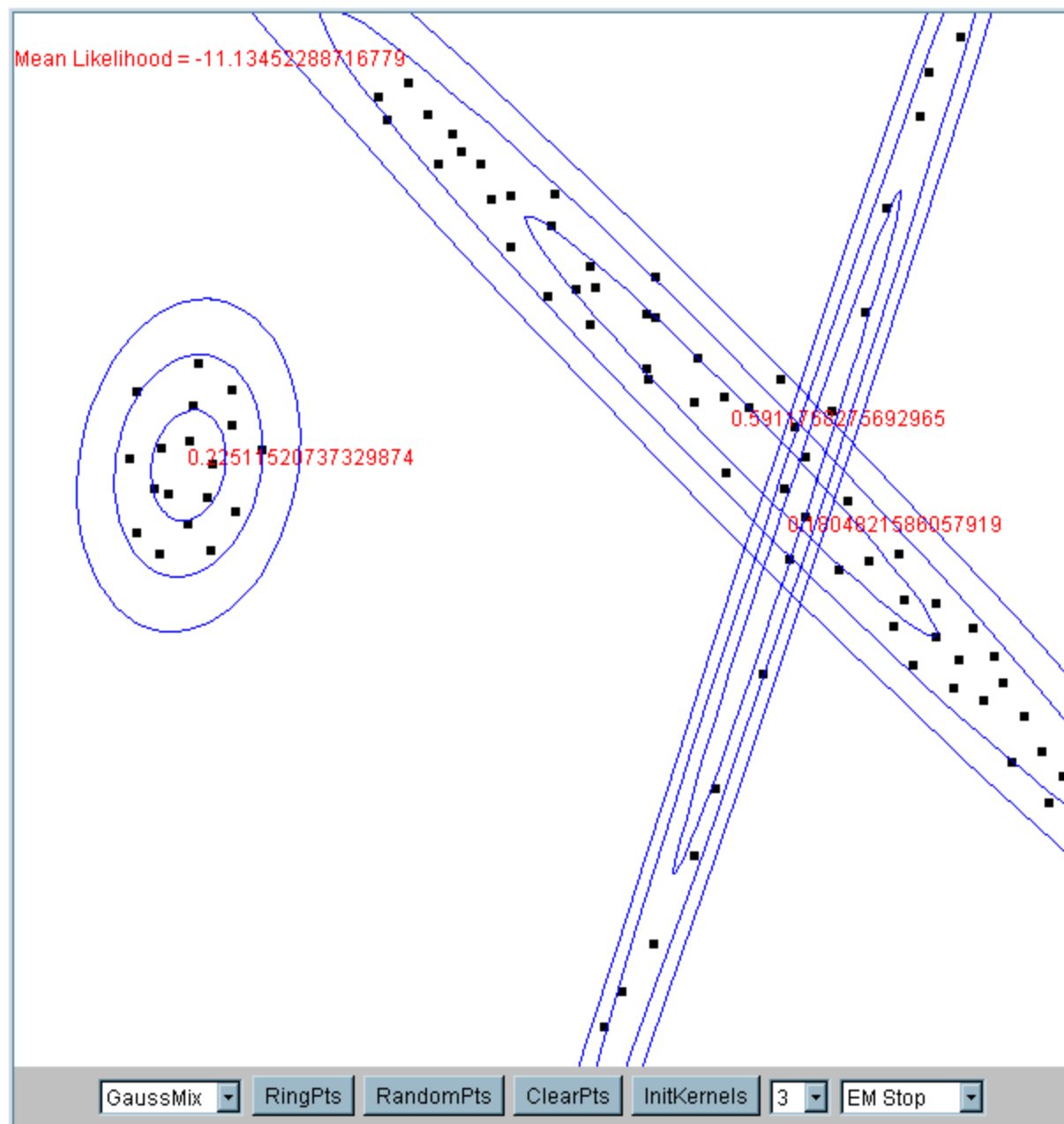
Iteration 2





Iteration 5

Iteration 25



Strength of Gaussian Mixture Models

- *Interpretability*: learns a generative model of each cluster
 - you can generate new data based on the learned model
- *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.
- Intuitive (?) objective function: optimizes data likelihood

Weakness of Gaussian Mixture Models

- Often terminates at a *local optimum*. Initialization is important.
- Need to specify K , the *number* of clusters, in advance
- Not suitable to discover clusters with *non-convex shapes*
- Summary
 - To learn Gaussian mixture, assign probabilistic membership based on current parameters, and re-estimate parameters based on current membership

Algorithm: K-means and GMM

1. Decide on a value for K , the number of clusters.
2. Initialize the K cluster centers / parameters (randomly).

K-means

GMM

3. Decide the class memberships of the N objects by assigning them to the nearest cluster center.
4. Re-estimate the K cluster centers, by assuming the memberships found above are correct.

3. E-step: assign *probabilistic* membership
4. M-step: re-estimate parameters based on *probabilistic* membership

5. Repeat 3 and 4 until parameters do not change.

Clustering methods: Comparison

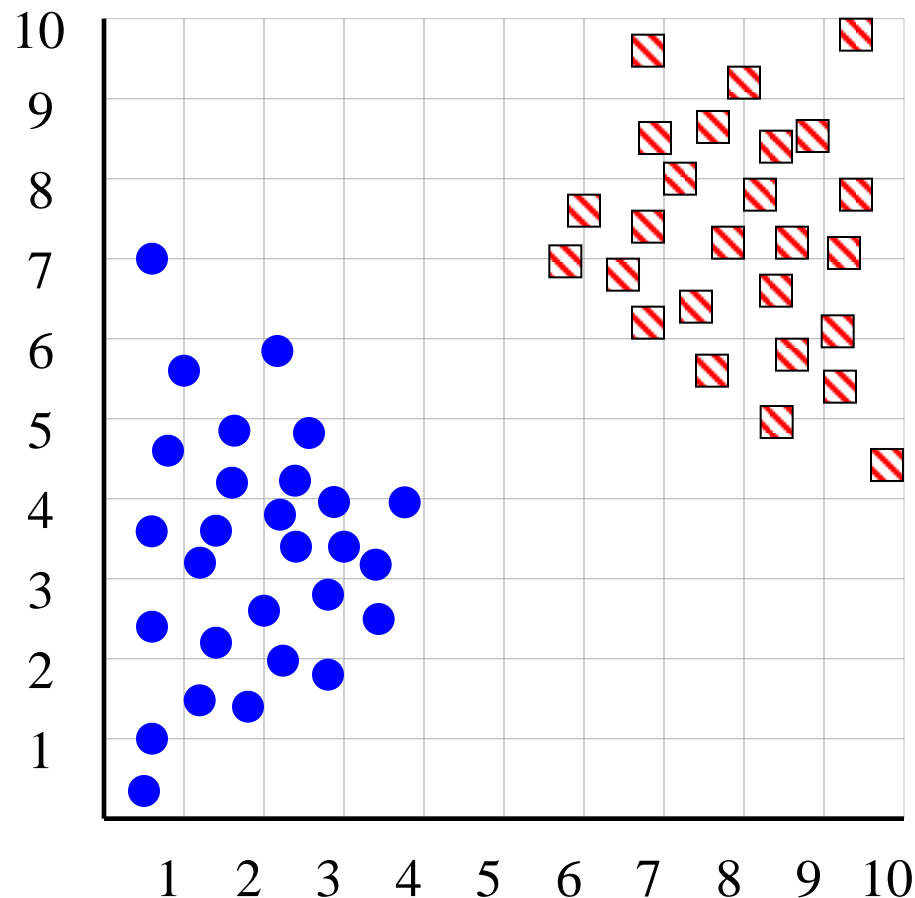
	Hierarchical	K-means	GMM
Running time	naively, $O(N^3)$	fastest (each iteration is linear)	fast (each iteration is linear)
Assumptions	requires a similarity / distance measure	strong assumptions	strongest assumptions
Input parameters	none	K (number of clusters)	K (number of clusters)
Clusters	subjective (only a tree is returned)	exactly K clusters	exactly K clusters

Outline

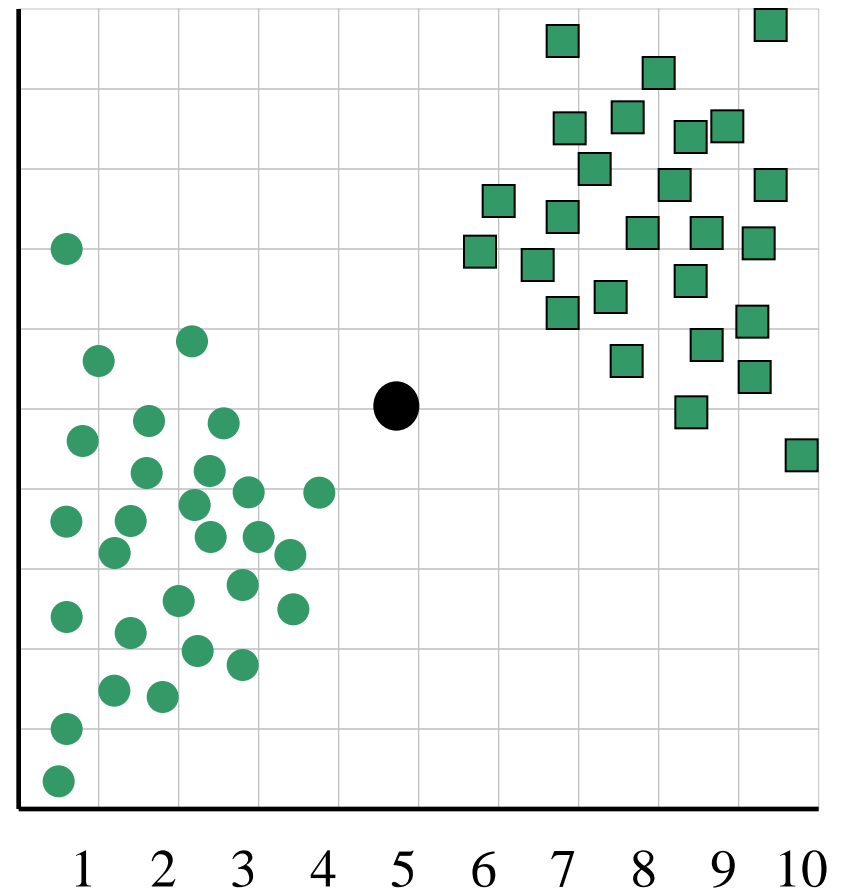
- Motivation
- Distance measure
- Hierarchical clustering
- Partitional clustering
 - K-means
 - Gaussian Mixture Models
 - Number of clusters

How can we tell the *right* number of clusters?

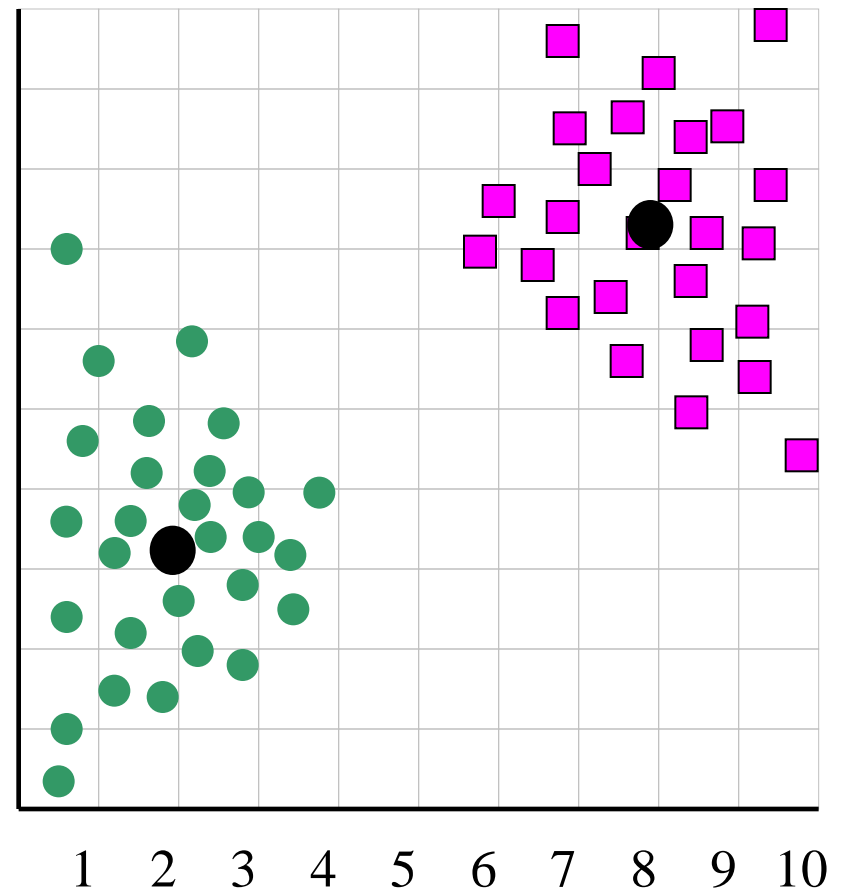
In general, this is an unsolved problem. However there are many approximate methods. In the next few slides we will see an example.



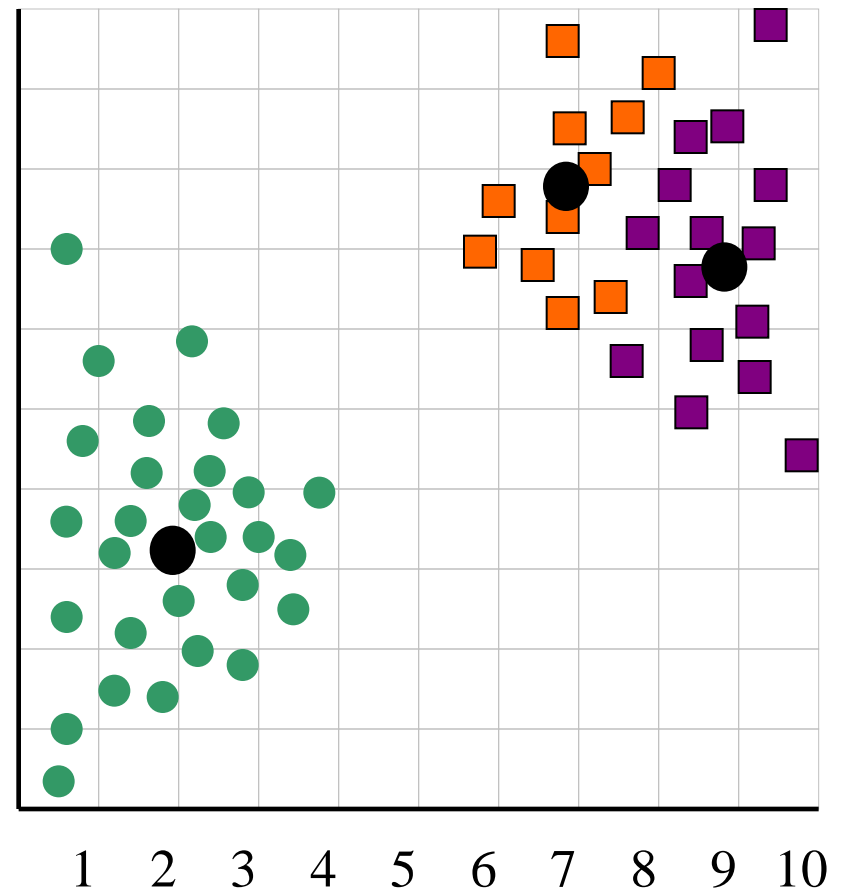
When $k = 1$, the objective function is 873.0



When $k = 2$, the objective function is 173.1

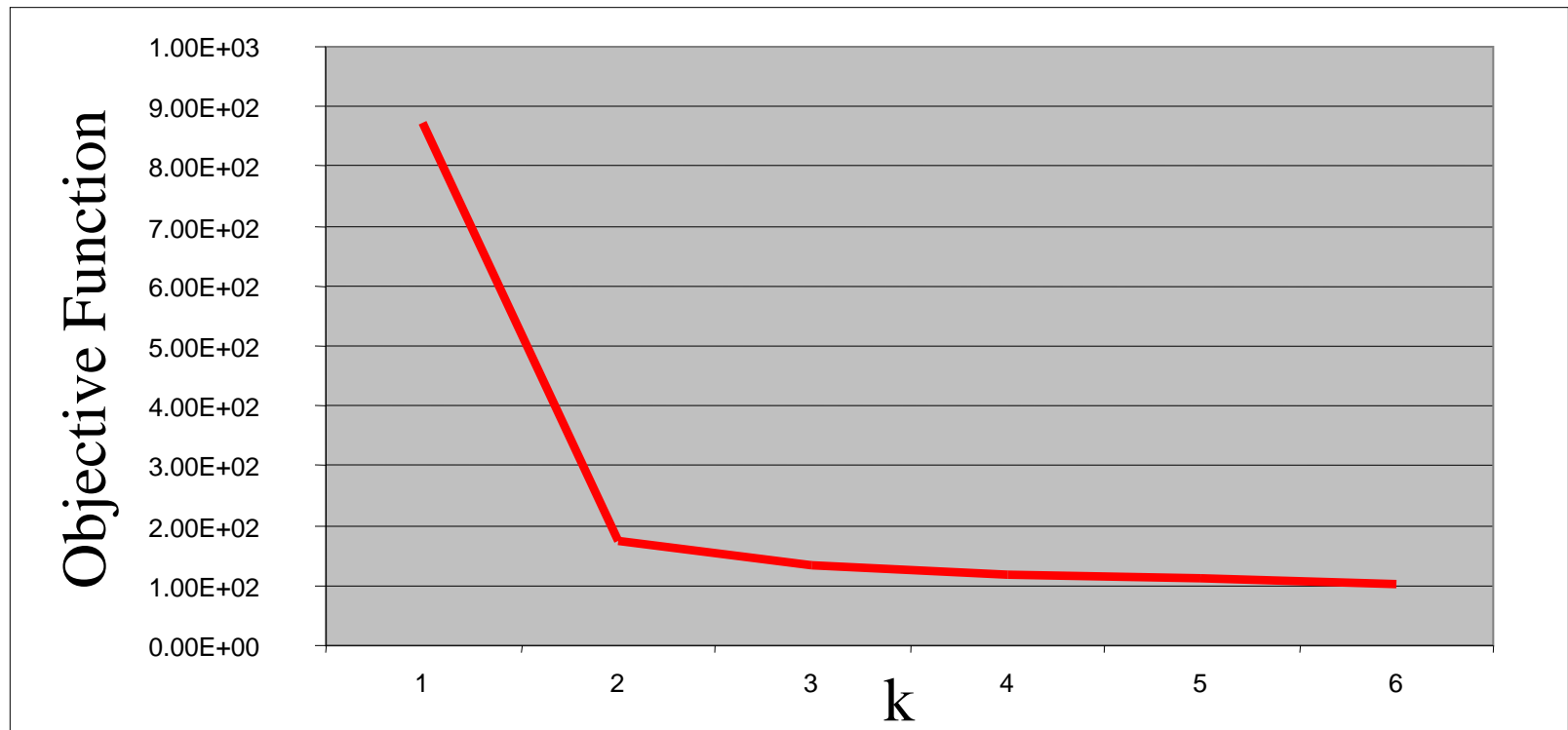


When $k = 3$, the objective function is 133.6



We can plot the objective function values for k equals 1 to 6...

The abrupt change at $k = 2$, is highly suggestive of two clusters in the data. This technique for determining the number of clusters is known as “knee finding” or “elbow finding”.

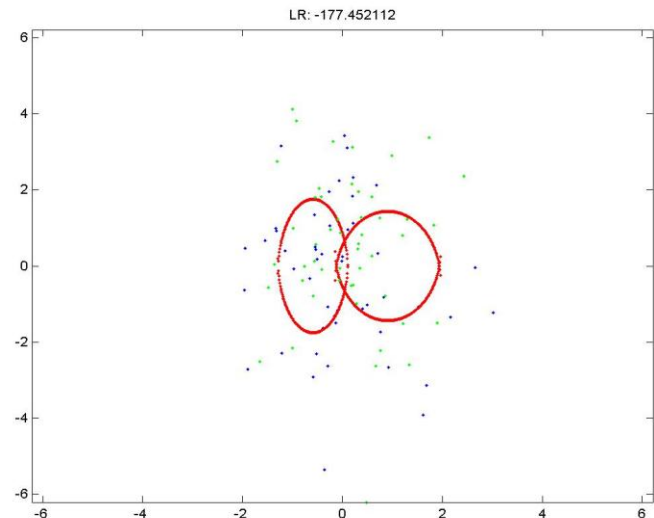
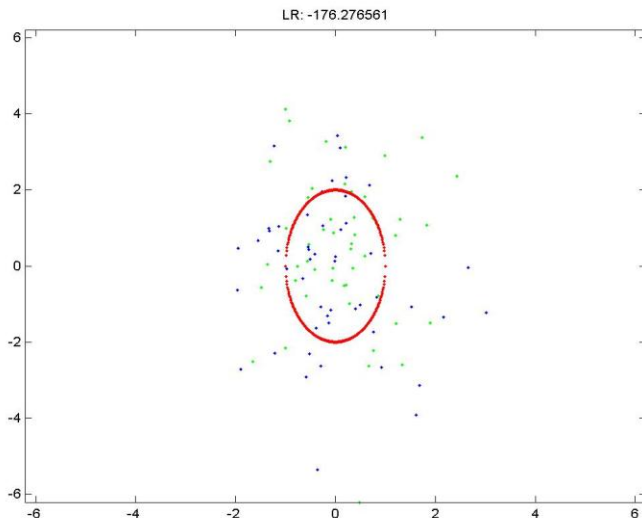


Note that the results are not always as clear cut as in this toy example

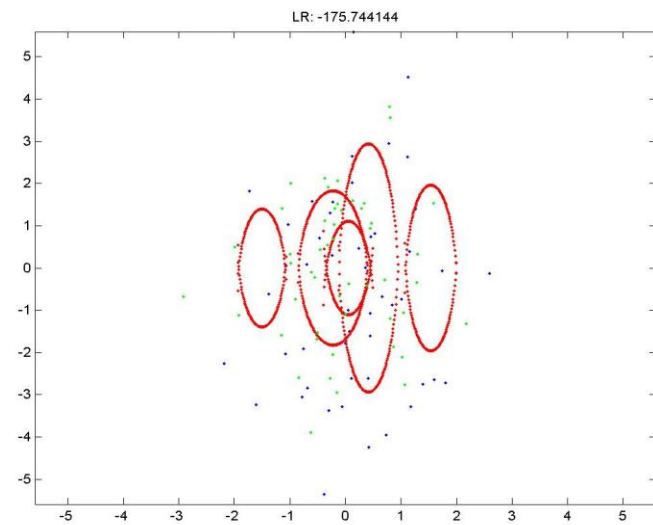
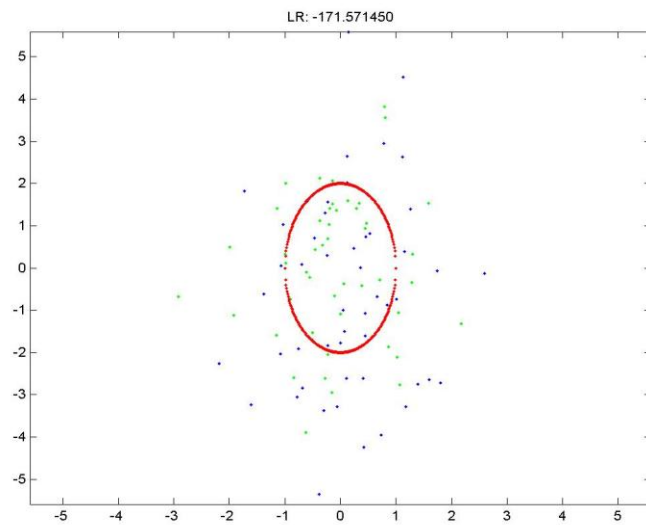
Cross validation

- We can also use cross validation to determine the correct number of classes
- Recall that GMMs is a generative model. We can compute the likelihood of the left out data to determine which model (number of clusters) is more accurate

$$p(x_1 \cdots x_n \mid \theta) = \prod_{j=1}^n \left(\sum_{i=1}^k p(x_j \mid C = i) w_i \right)$$



Cross validation

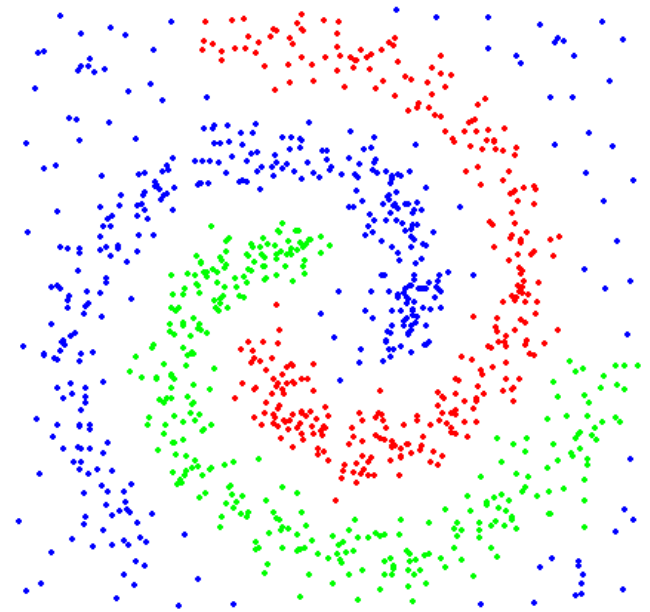


Other clustering methods (briefly)

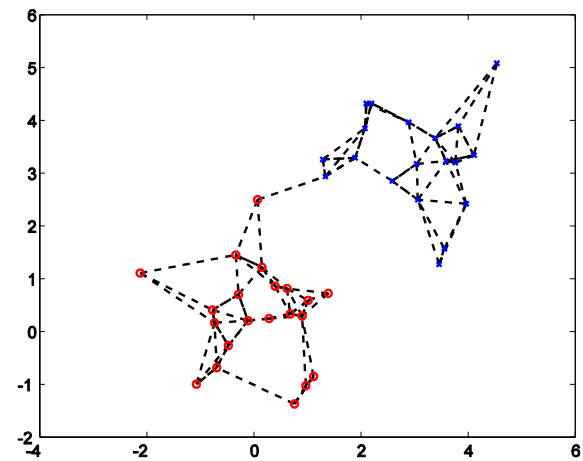
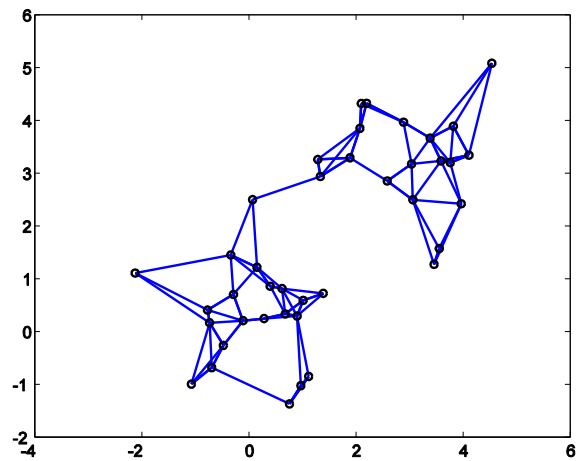
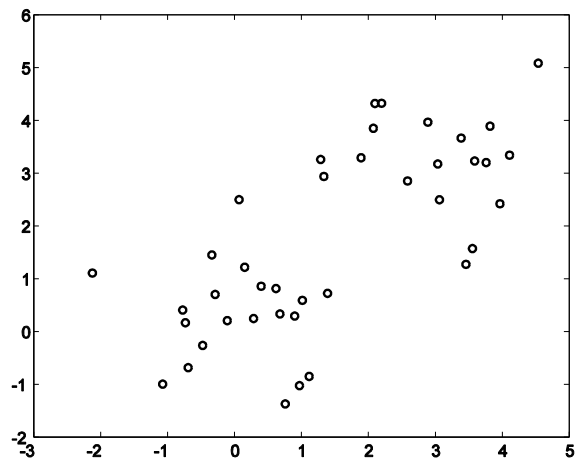
Graph based clustering

- Represents points as a graph with edges corresponding to distance / similarity
- Apply graph based segmentation algorithms to identify clusters (looking for strongly connected components)
- Also known as ‘spectral clustering’

How would k-means perform on this? Hierarchical clustering?

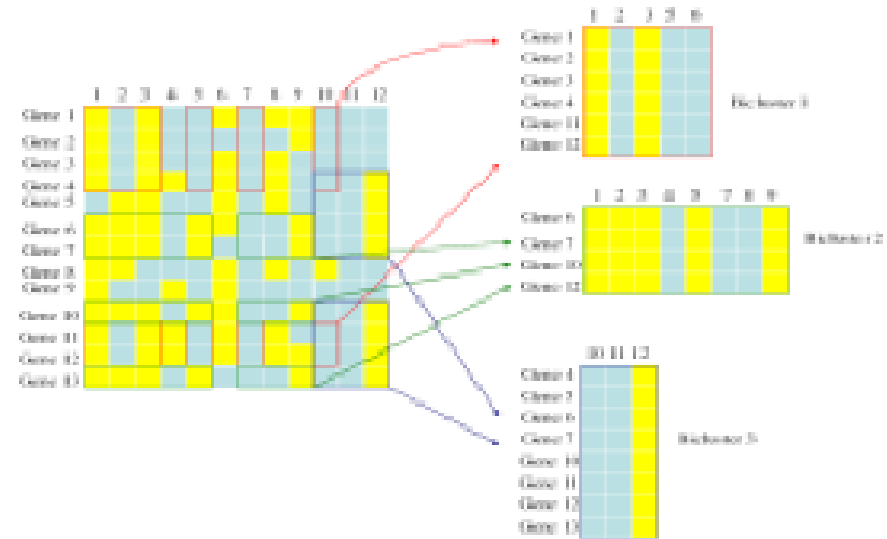


Example



BiClustering

- What if we have objects that belong to more than one class?
 - Tony Parker's new rap album
 - A gene that is involved in cancer and Parkinsons disease
- BiClustering algorithms find a subset of the input features (words, gene levels etc.) and cluster based on these features
- Other (partially overlapping) subsets may be associated with other clusters



Constrained clustering

- In many cases we have prior information that we can use to improve or inform the clustering algorithm
 - ‘These documents should not be co-clustered’
 - ‘Both of these genes are associated with cancer’
- The information can be either ‘hard’ (have to be in the same cluster) or ‘soft’ (probability in the same cluster)
- Some algorithms can use this information to improve the outcome
- Often done by revising the target function (which usually means that it requires a new search strategy)

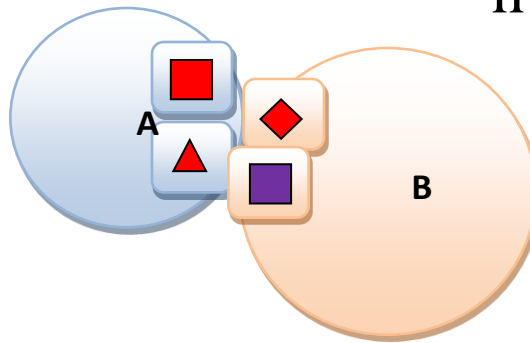
Clustering data when knowledge about co-membership is provided

$$h^*(x) = \operatorname{argmin}_h D(C_h, x)$$

$$\min_i \sum D(x_i, C_{h^*(x_i)})$$

$h^*(x)$ - x 's cluster

C_h - Center for cluster
 h



Constrained clustering: A new target function

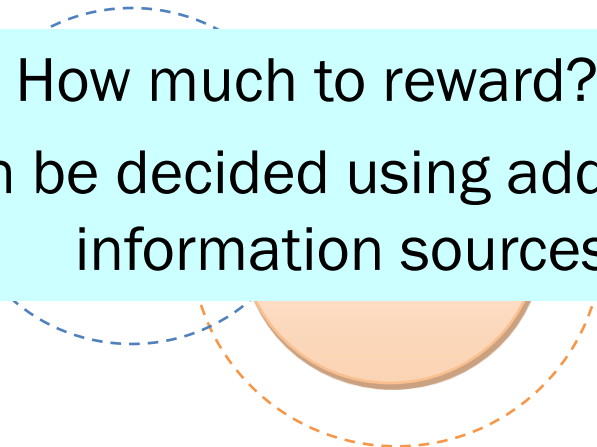
- We want to use both, text data and prior information
- How? Changing the objective function of the algorithm to use side information.

$$\min_{I(x_1, x_2, x_3)} \sum \left[W_{ort}(I) + \sum_j D(x_j, C_{h^*(x_j)}) \right]$$

Input sets

Penalty for partitioning objects that should likely be combined

How much to reward?
Can be decided using additional information sources



What you should know

- Why is clustering useful
- What are the different types of clustering algorithms
- What are the assumptions we are making for each, and what can we get from them
- Unsolved issues: number of clusters, initialization, etc.