Expectation-Maximization (EM)

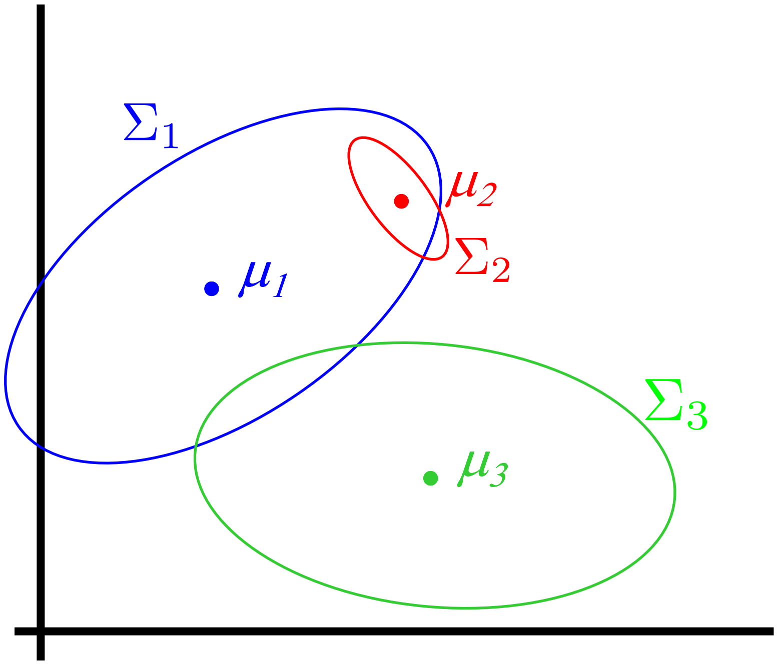
Aarti Singh

Machine Learning 10-315
Nov 23, 2020

Some slides courtesy of Eric Xing, Carlos Guestrin

Mixture models (Gaussian)

$$x_1, \ldots, x_m \sim p(x) = \sum_{i=1}^{k} \underbrace{p(x|Y=i)}_{\text{Mixture component}} \underbrace{P(Y=i)}_{\text{Mixture proportion, } p_i}$$



Gaussian mixture model

$$p(x|Y=i) \sim \mathcal{N}(\mu_i, \Sigma_i)$$

Parameters: $\{p_i, \mu_i, \Sigma_i\}_{i=1}^{K}$

- How to estimate parameters? Max Likelihood
  But don’t know labels Y (recall Gaussian Bayes classifier)

Expectation-Maximization (EM)

A general algorithm to deal with hidden data, but we will study it in the context of unsupervised learning (hidden labels)

- No need to choose step size as in Gradient methods.
- EM is an Iterative algorithm with two linked steps:

  E-step: fill-in hidden data (Y) using inference

  M-step: apply standard MLE/MAP method to estimate parameters

  $\{p_i, \mu_i, \Sigma_i\}^k_{i=1}$

- This procedure monotonically improves the marginal likelihood (or leaves it unchanged). Thus it always converges to a local optimum of the likelihood.

EM for spherical, same variance GMMs same mixture proportions

Initialize: $\mu_1, \mu_2, \ldots, \mu_K$ randomly

E-step

Compute "expected" classes of all datapoints for each class

$$P(y=i|x_j,\mu_1\ldots\mu_k) \propto \exp\left(-\frac{1}{2\sigma^2}\|x_j-\mu_i\|^2\right)P(y=i)$$

In K-means "E-step" we do hard assignment

EM does soft assignment

M-step

Compute Max. like **μ** given our data's class membership distributions (weights)

$$\mu_i = \frac{\sum_{j=1}^{m} P(y=i|x_j)x_j}{\sum_{j=1}^{m} P(y=i|x_j)}$$

Exactly same as MLE with weighted data

Iterate.

EM for general GMMs

Iterate. On iteration t let our estimates be

$\lambda_t = \{ \mu_1^{(t)}, \mu_2^{(t)} \dots \mu_k^{(t)}, \Sigma_1^{(t)}, \Sigma_2^{(t)} \dots \Sigma_k^{(t)}, p_1^{(t)}, p_2^{(t)} \dots p_k^{(t)} \}$

$p_i^{(t)}$ is shorthand for estimate of *P(y=i)* on t'th iteration

E-step

Compute "expected" classes of all datapoints for each class

$$\mathrm{P}\left(y = i \middle| x_j, \lambda_t\right) \propto p_i^{(t)} \mathrm{p}\left(x_j \middle| \mu_i^{(t)}, \Sigma_i^{(t)}\right)$$

Just evaluate a Gaussian at x_j

M-step

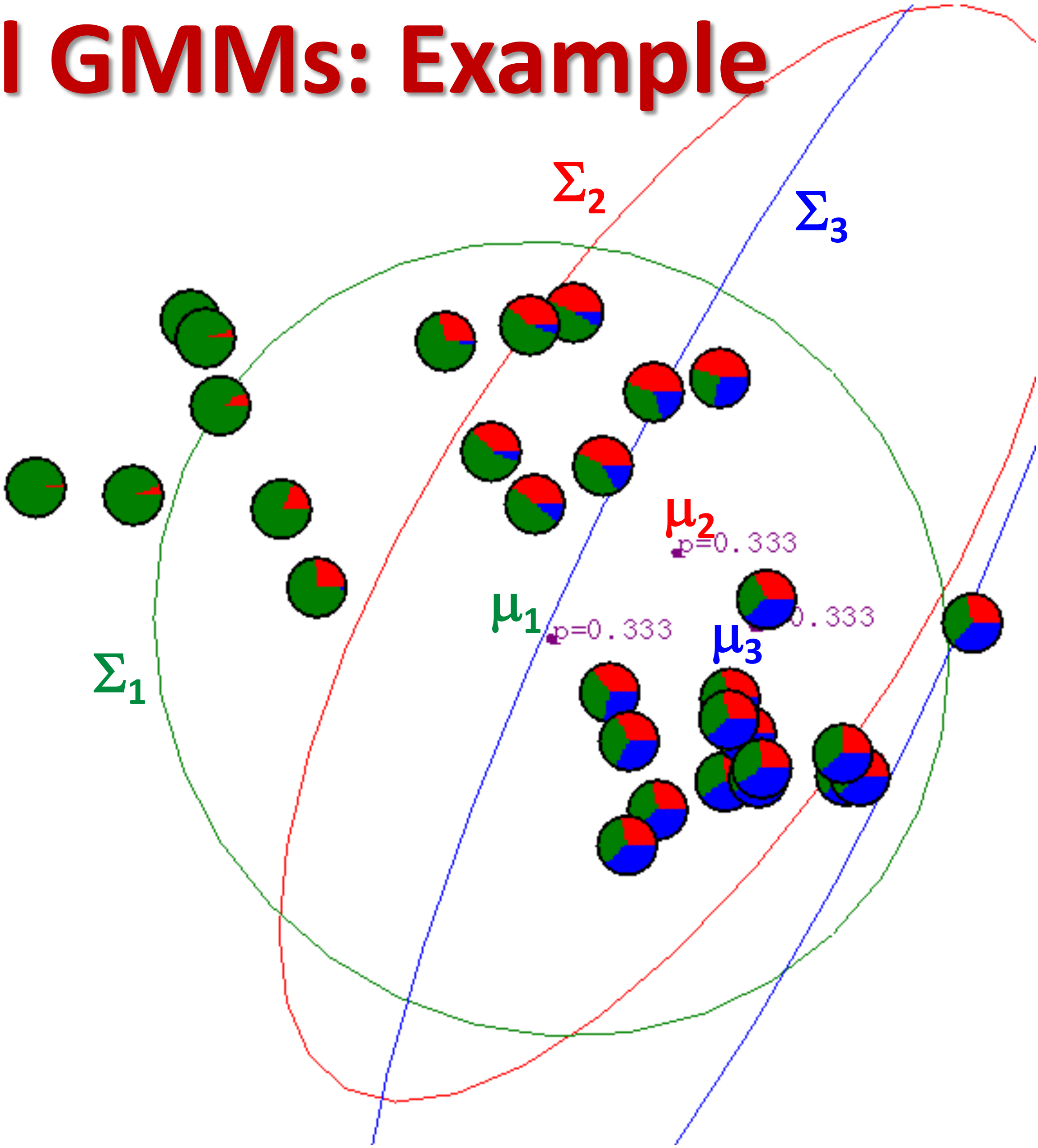
Compute MLEs given our data's class membership distributions (weights)

$$\mu_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right) x_j}{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right)} \qquad \Sigma_\mathrm{i}^{(t+1)} = \frac{\sum_\mathrm{j} \mathrm{P}\left(\mathrm{y} = \mathrm{i} \middle| \mathrm{x}_\mathrm{j}, \lambda_\mathrm{t}\right)\left(\mathrm{x}_\mathrm{j} - \mu_\mathrm{i}^{(t+1)}\right)\left(\mathrm{x}_\mathrm{j} - \mu_\mathrm{i}^{(t+1)}\right)^\mathrm{T}}{\sum_\mathrm{j} \mathrm{P}\left(\mathrm{y} = \mathrm{i} \middle| \mathrm{x}_\mathrm{j}, \lambda_\mathrm{t}\right)}$$

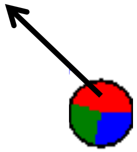
$$p_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right)}{m}$$

m = #data points

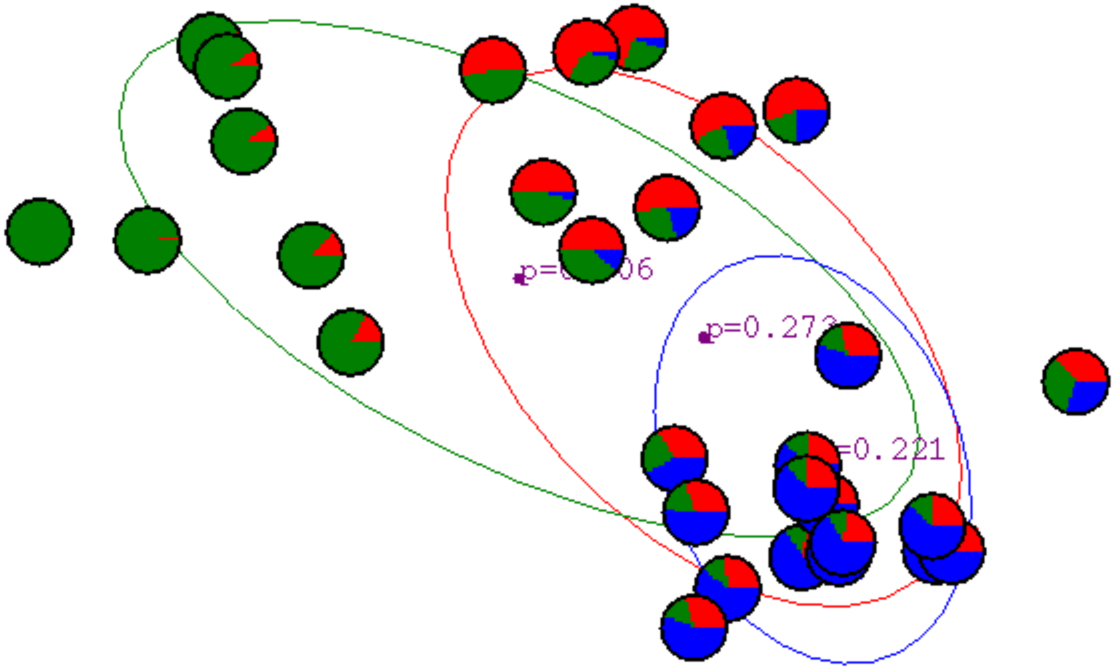
EM for general GMMs: Example



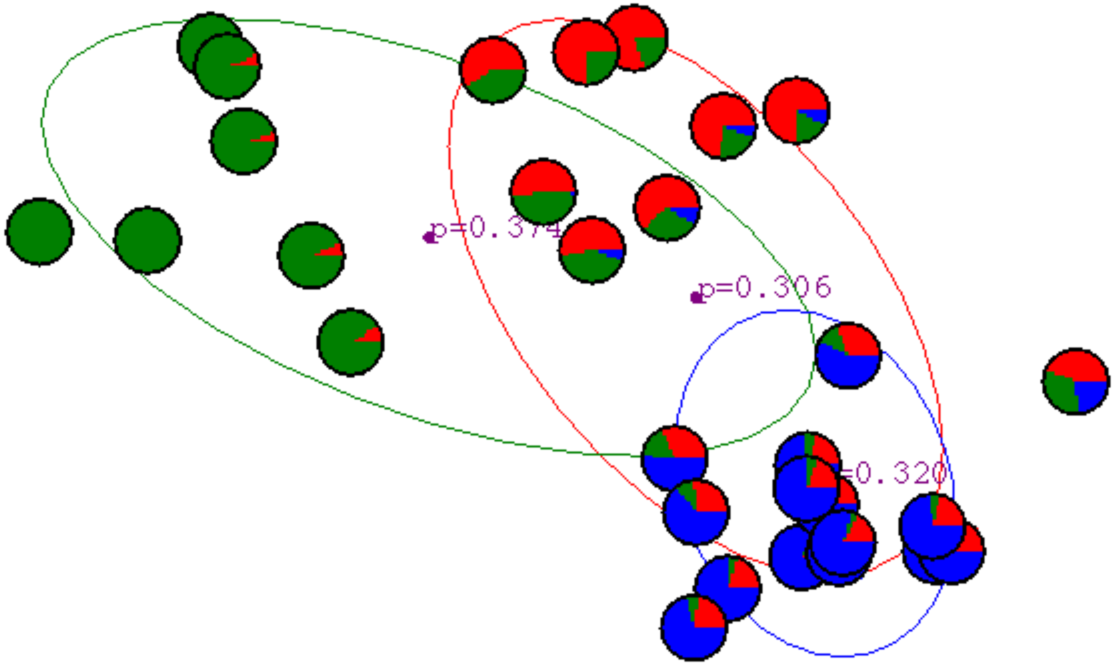
$P(y = \bullet \mid x_j,\mu_1,\mu_2,\mu_3,\Sigma_1,\Sigma_2,\Sigma_3,p_1,p_2,p_3)$



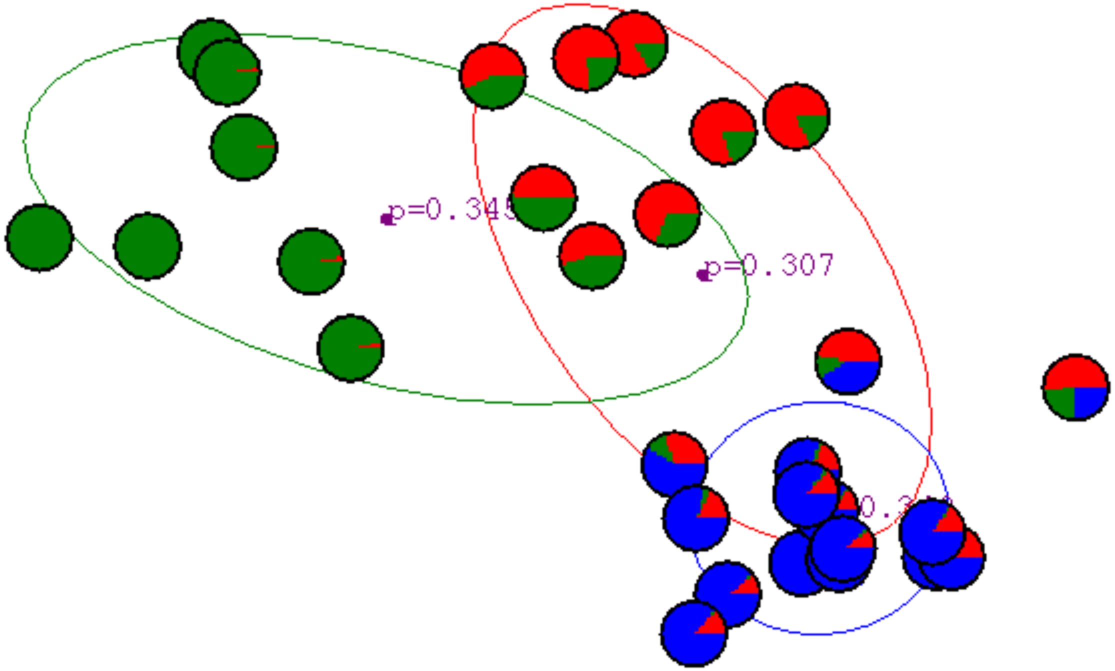
After 1^{st} iteration



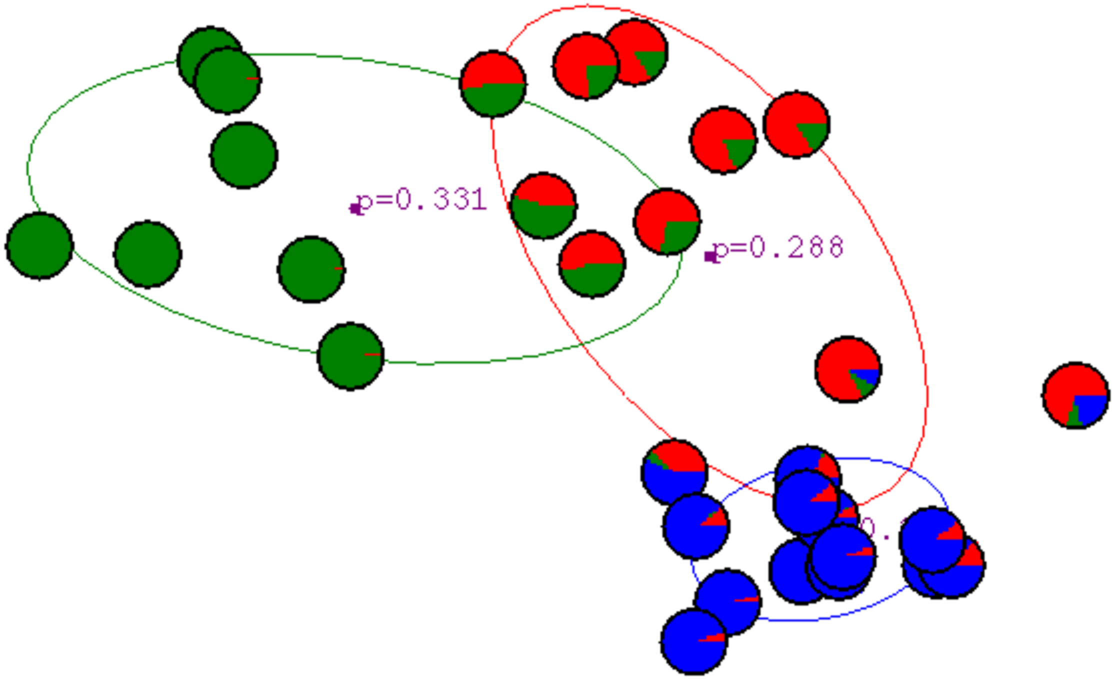
After 2nd iteration



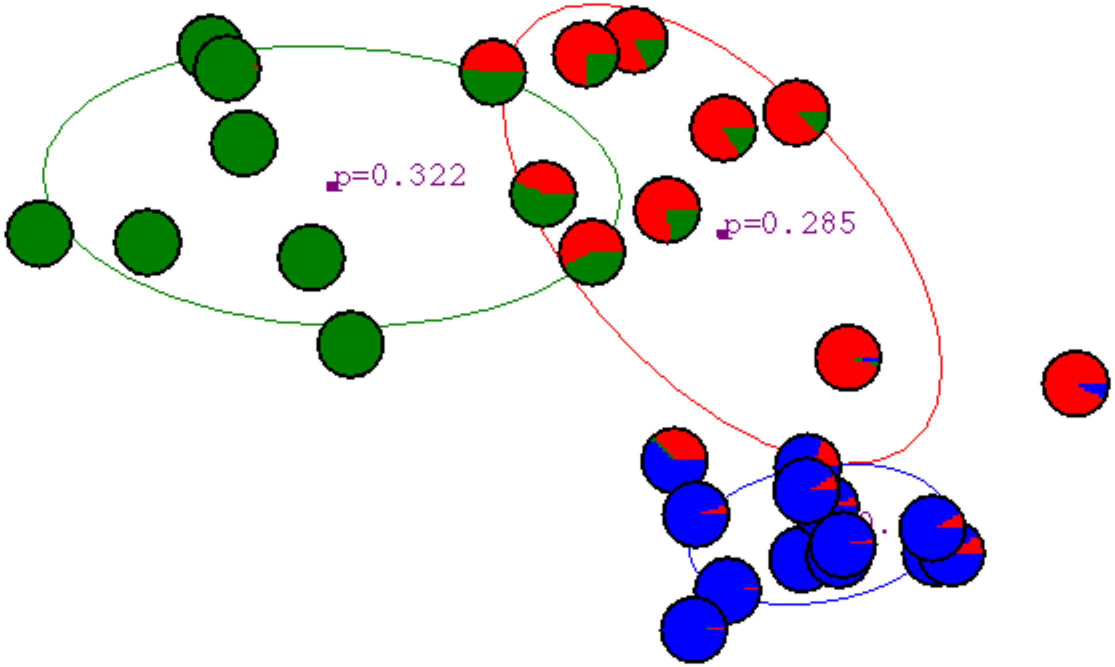
After 3rd iteration



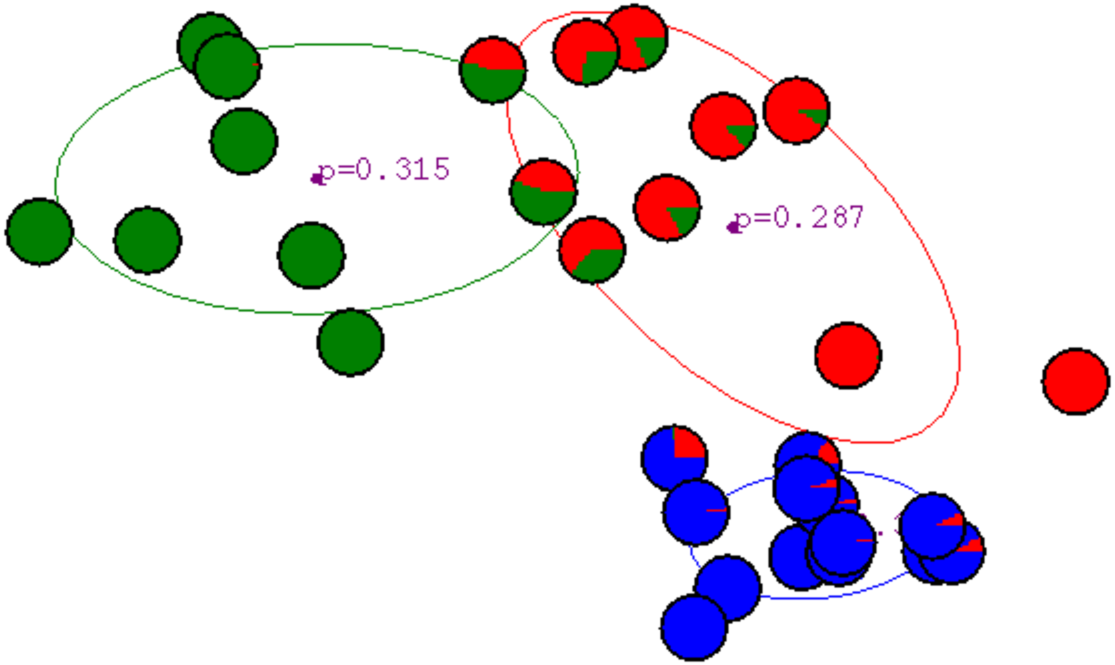
After 4th iteration



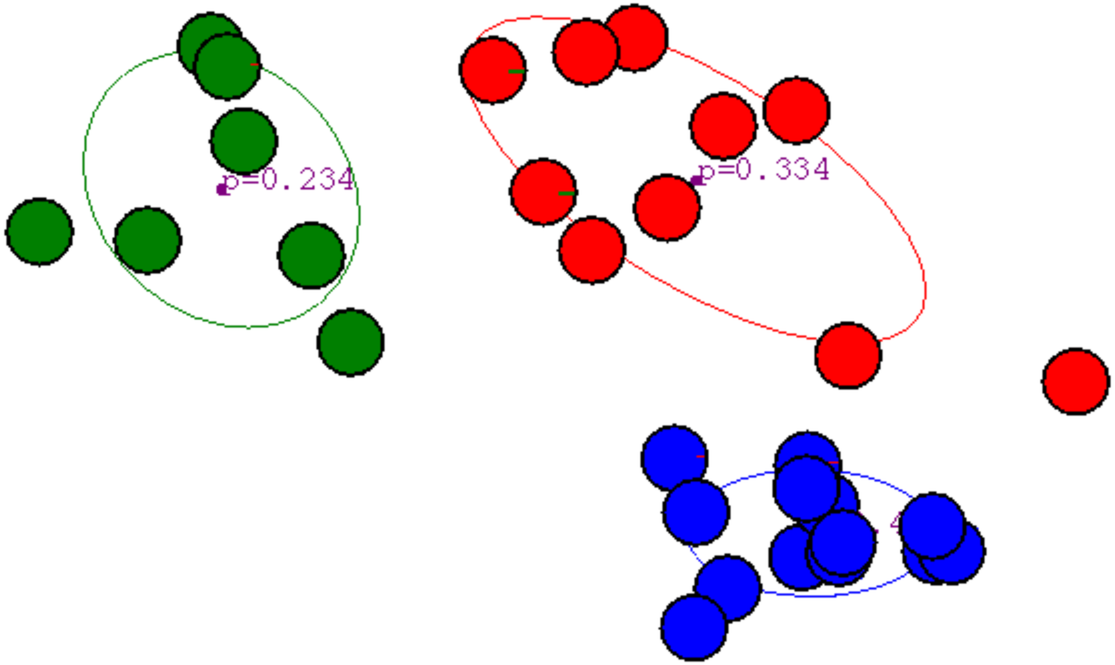
After 5th iteration



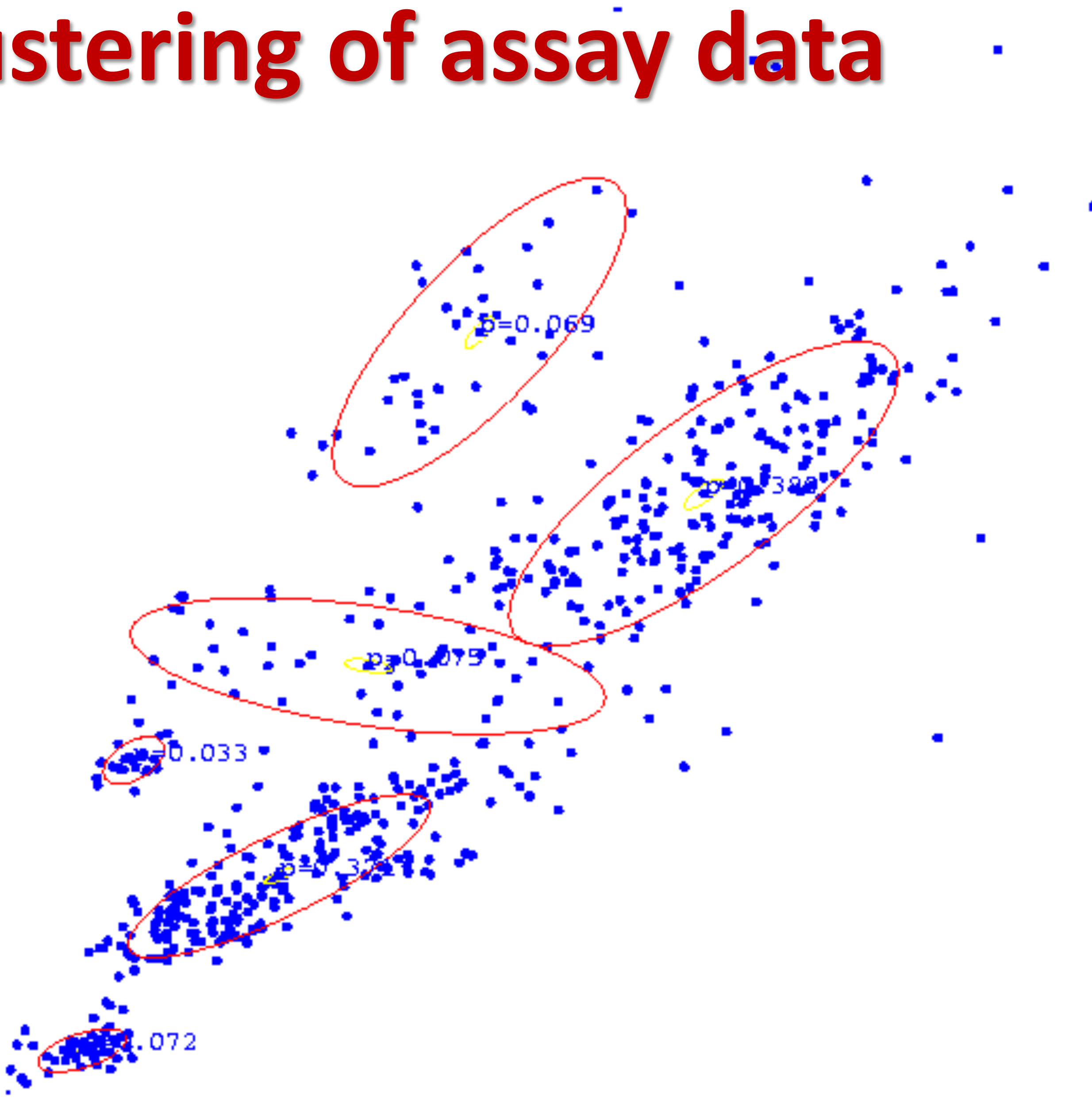
After 6th iteration



After 20th iteration



GMM clustering of assay data



General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

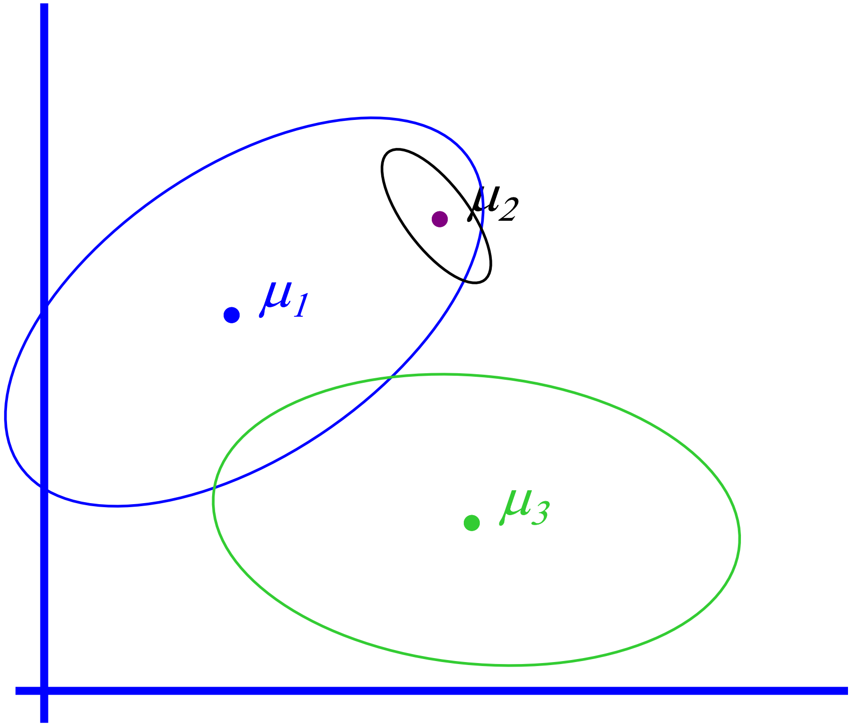
$$p(x) = \sum_i p(x|y=i)\, P(y=i)$$

$$\underbrace{p(x|y=i)}_{\text{Mixture component}} \quad \underbrace{P(y=i)}_{\text{Mixture proportion}}$$

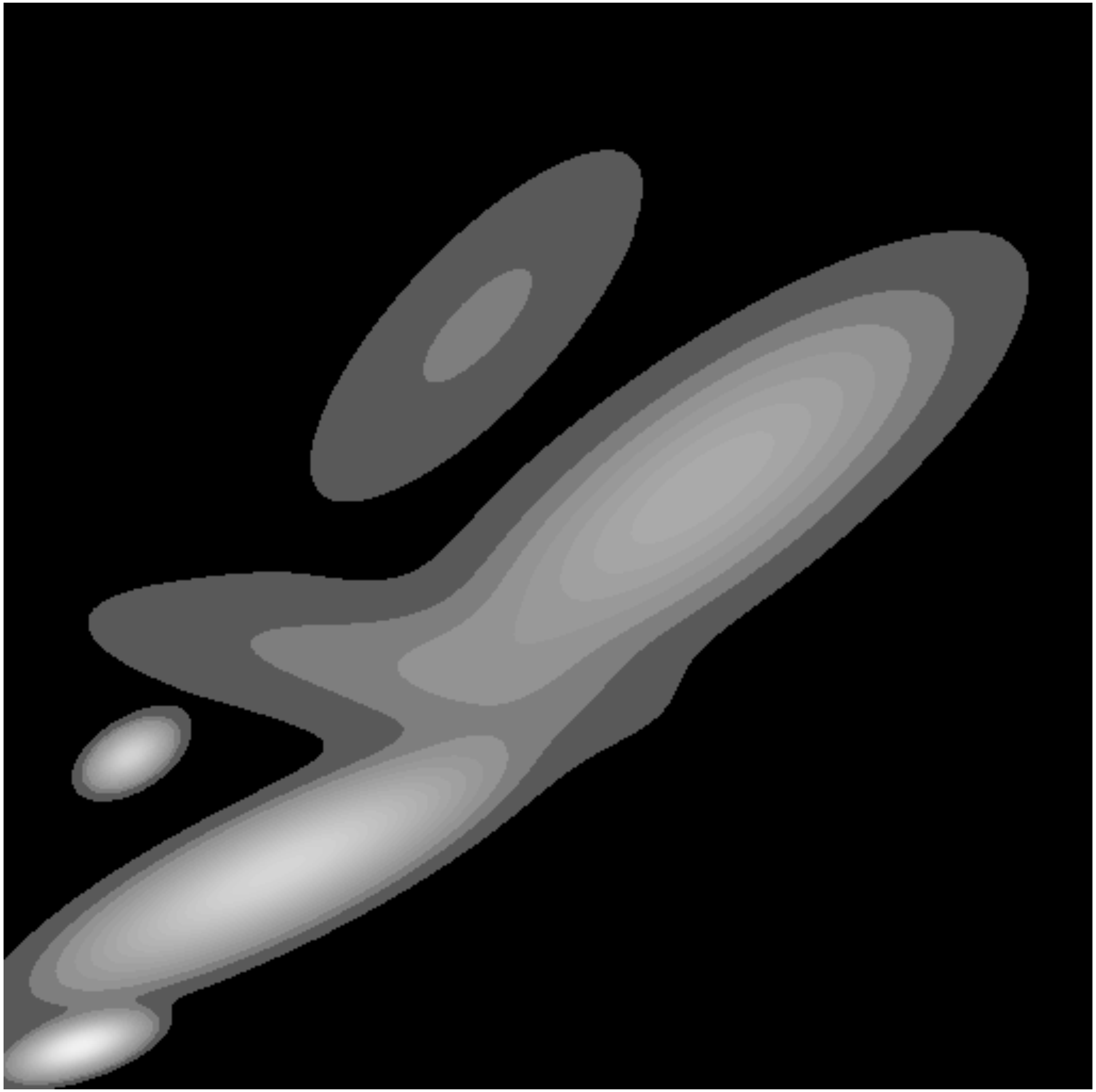
Mixture component

Mixture proportion

$$p(x|y=i) \sim \mathrm{N}(\mu_i, \Sigma_i)$$

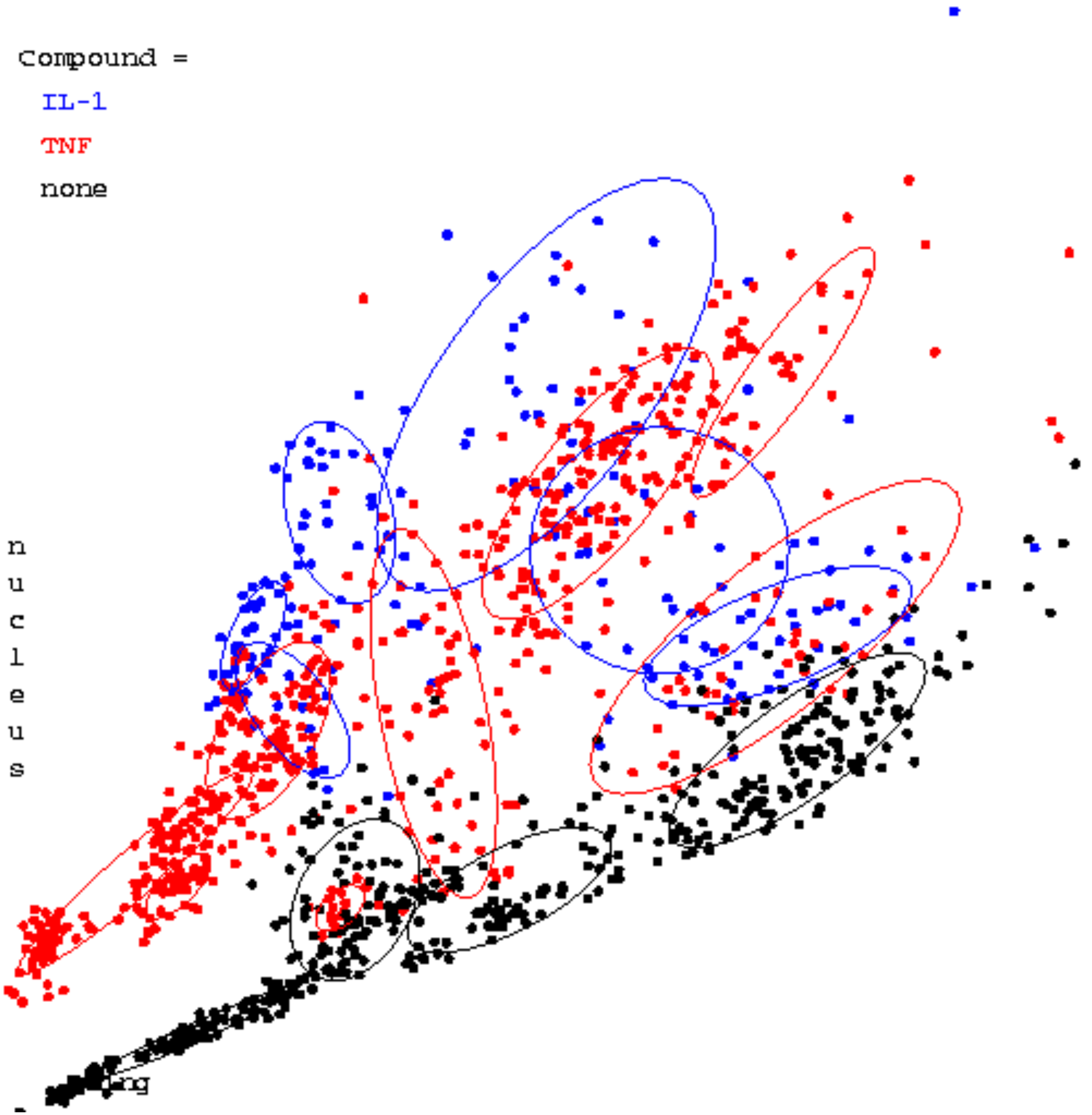


Resulting Density Estimator

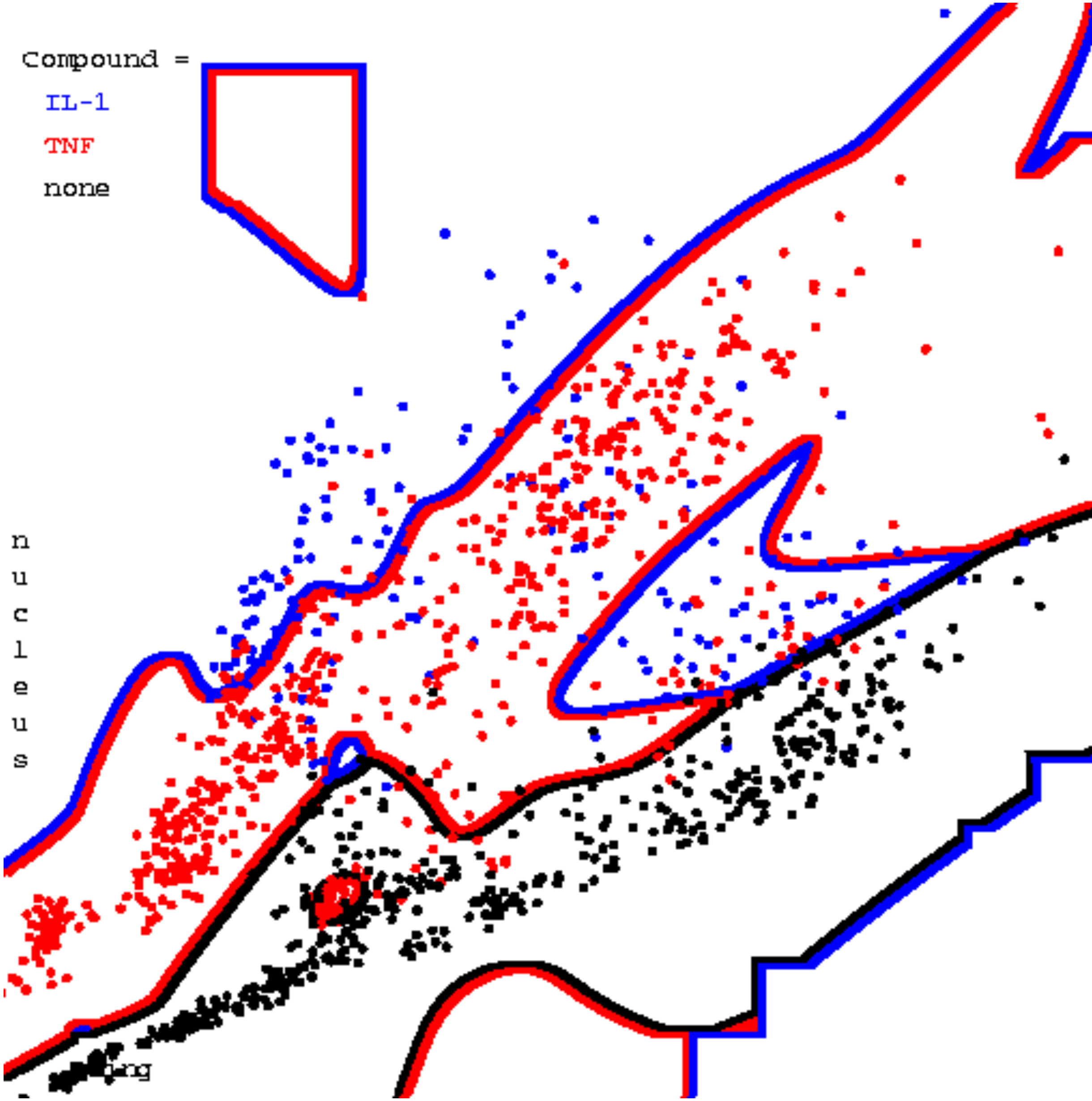


Three classes of assay

(each learned with it's own mixture model)



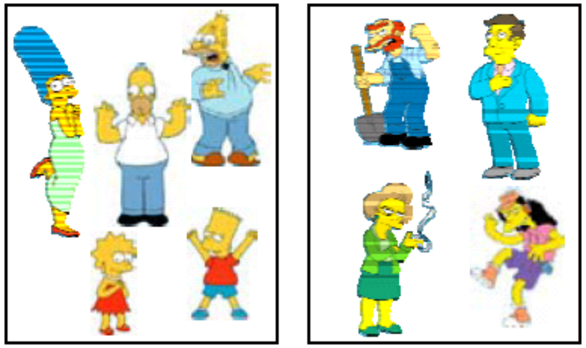
Resulting Bayes Classifier



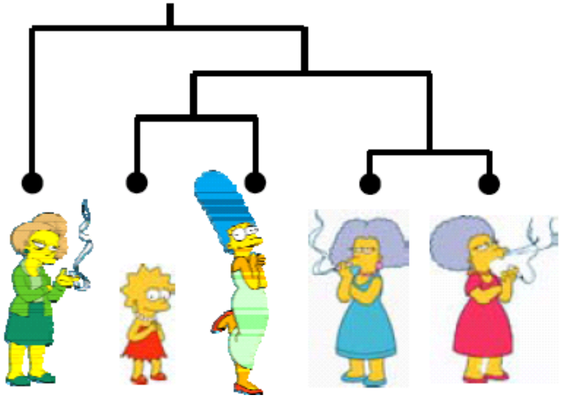
Summary: EM Algorithm

- A way of maximizing likelihood function for hidden variable models. Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces:
  1. Estimate some “missing” or “unobserved” data from observed data and current parameters.
  2. Using this “complete” data, find the maximum likelihood parameter estimates.
- Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:
  1. E-step: soft cluster assignment for each data point
  2. M-step: update parameters of each mixture component
- EM can get stuck in local minima.
- BUT Extremely popular in practice.

Clustering Algorithms



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



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

Hierarchical Clustering

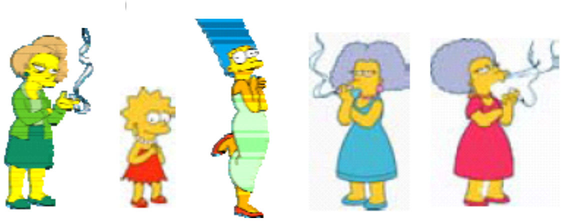
- Bottom-Up Agglomerative Clustering

  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



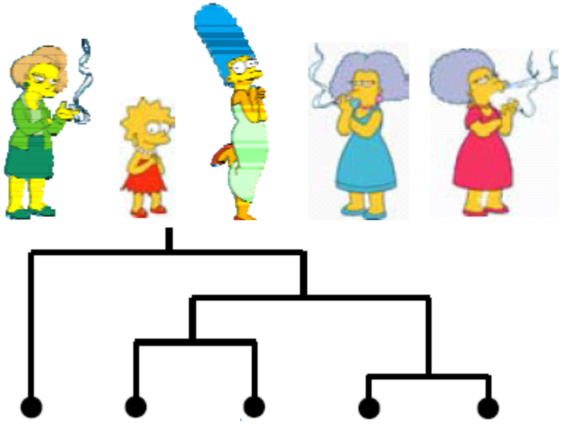
- Top-Down divisive

  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.

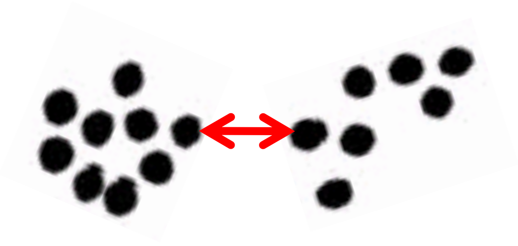
  More accurate but complex to implement



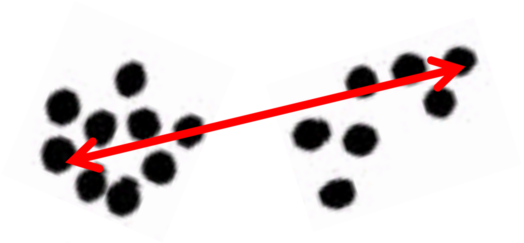
Bottom-up Agglomerative clustering

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

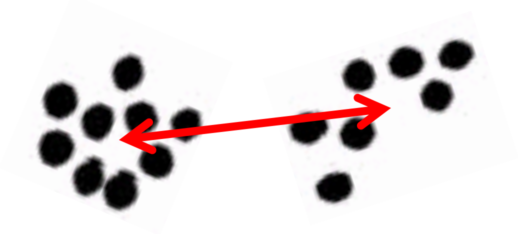
- Single-Linkage
  - Nearest Neighbor: similarity between their closest members.



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



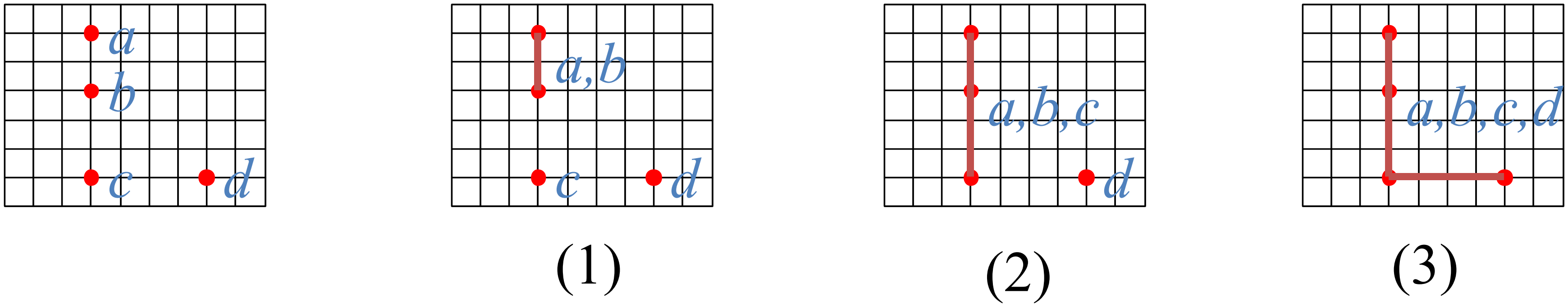
- Centroid
  - Similarity between the centers of gravity

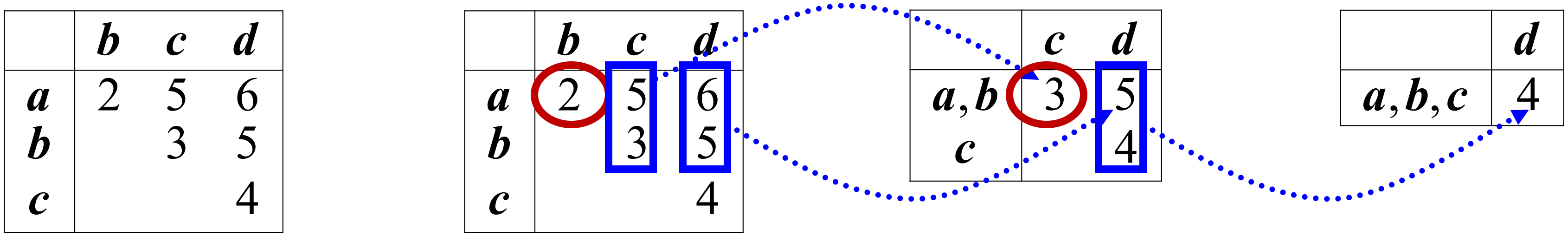


- Average-Linkage
  - Average similarity of all cross-cluster pairs.

Single-Linkage Method

Euclidean Distance





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

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

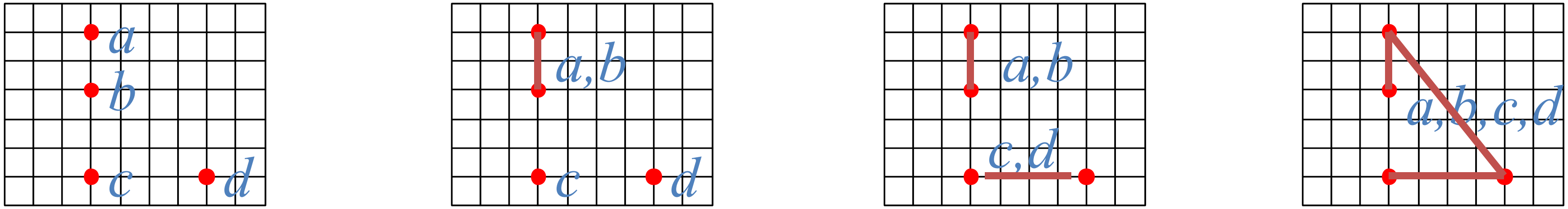
	c	d
a,b	3	5
c		4

	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

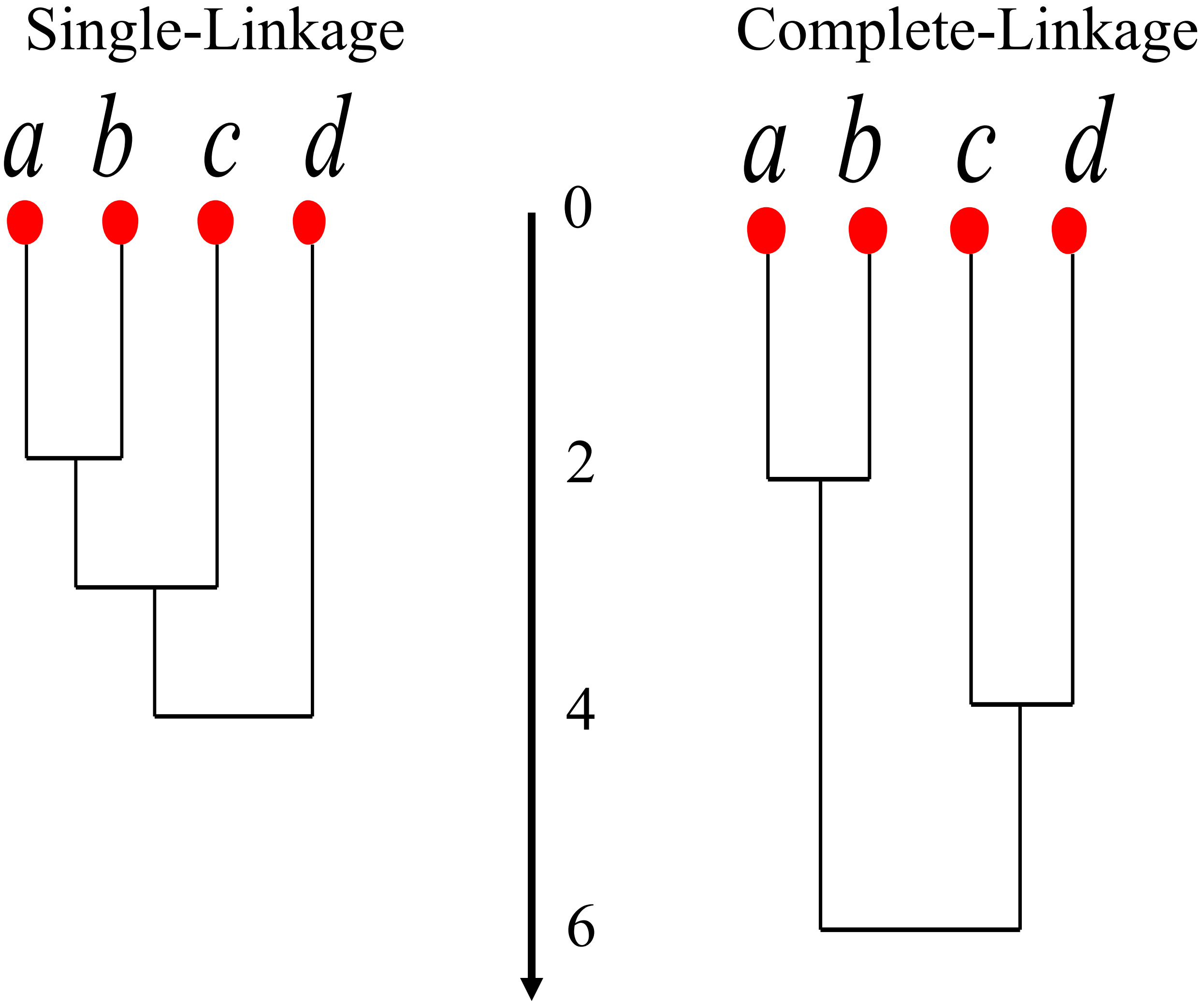
	b	**c**	**d**
a	2	5	6
b		3	5
c			4

	c	**d**
a,b	5	6
c		4

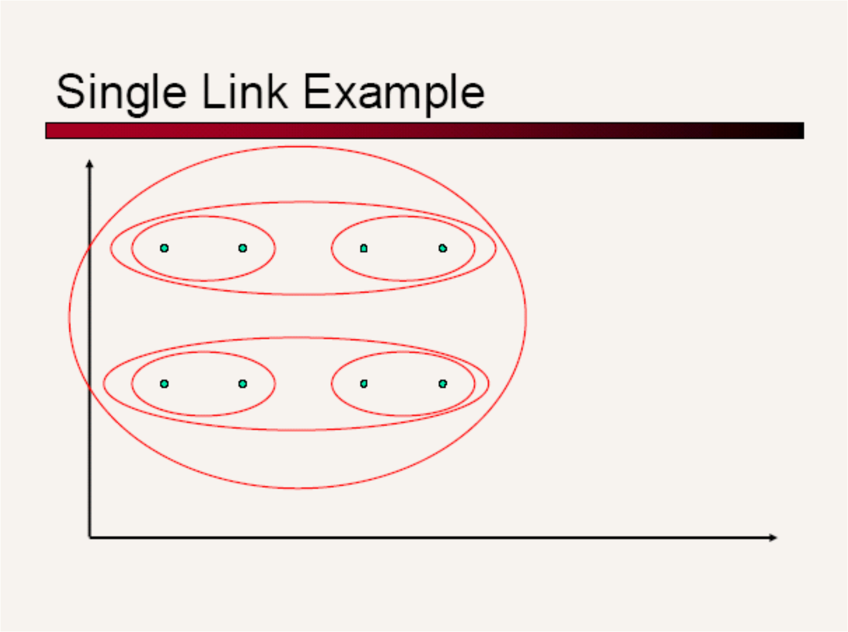
	c,d
a,b	6

Distance Matrix

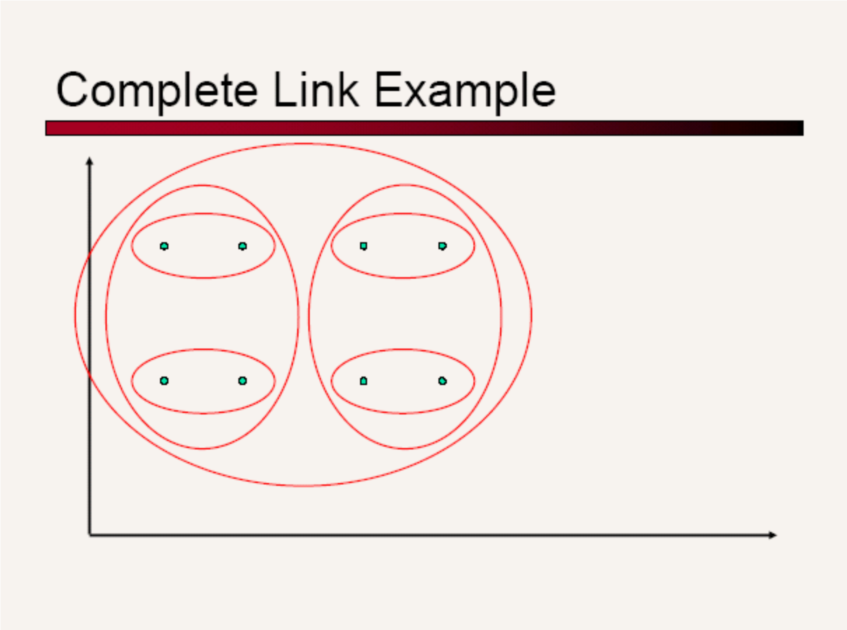
Dendrograms



Another Example



Single Link Example



Complete Link 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)

Computational Complexity (K-means)

- 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).

What you need to know…

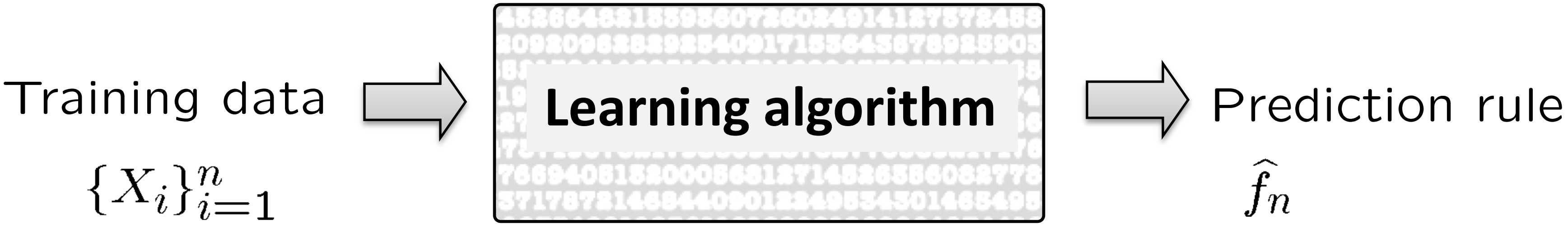
- Partition based clustering algorithms
  - K-means
    - Coordinate descent
    - Seeding
    - Choosing K
  - Mixture models

    EM algorithm

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

Unsupervised Learning

"Learning from unlabeled/unannotated data" (without supervision)



Training data

$\{X_i\}_{i=1}^n$

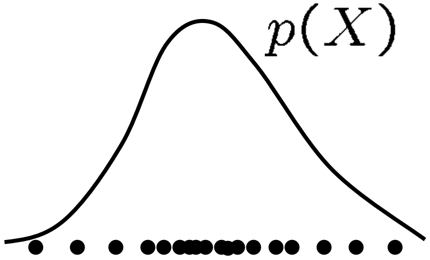
Learning algorithm

Prediction rule

$\widehat{f}_n$

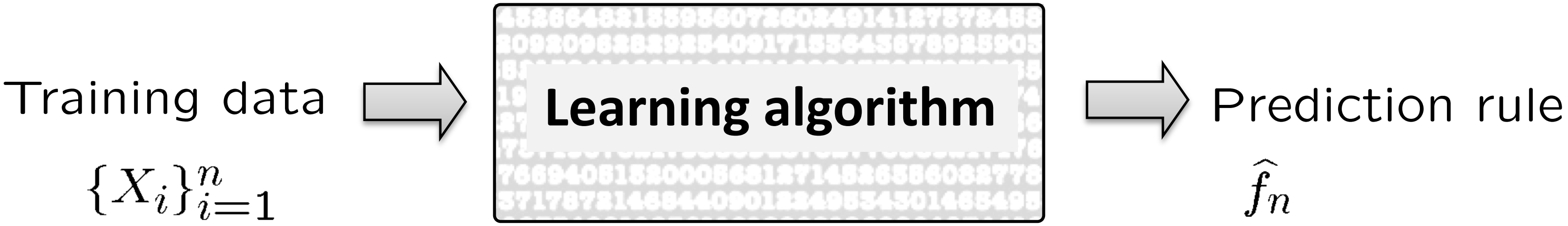
What can we predict from unlabeled data?

- Density estimation



Unsupervised Learning

"Learning from unlabeled/unannotated data" (without supervision)



Training data

$\{X_i\}_{i=1}^{n}$

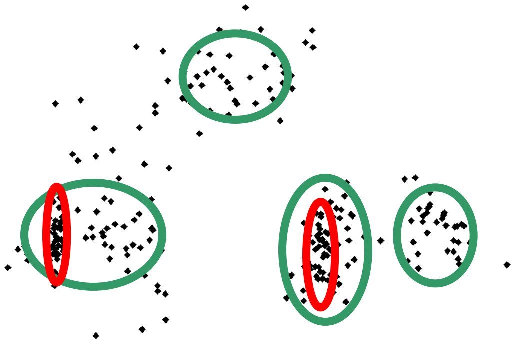
Learning algorithm

Prediction rule

$\widehat{f}_n$

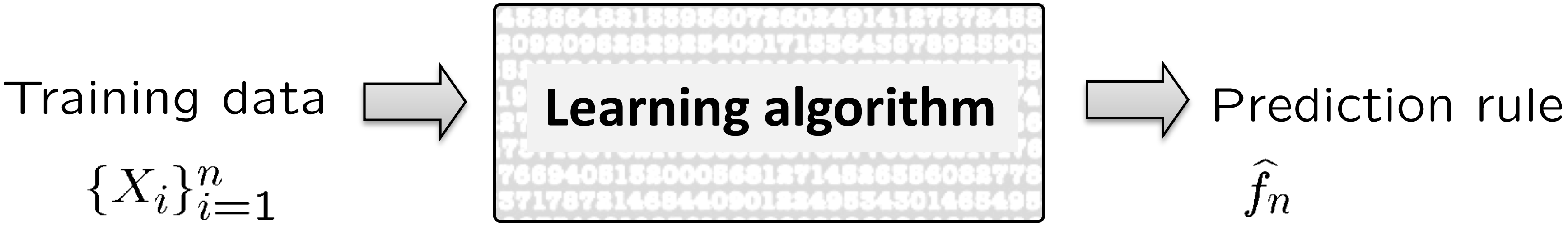
What can we predict from unlabeled data?

- Density estimation
- Groups or clusters in the data



Unsupervised Learning

"Learning from unlabeled/unannotated data" (without supervision)



Training data

$\{X_i\}_{i=1}^n$

Learning algorithm

Prediction rule

$\widehat{f}_n$

What can we predict from unlabeled data?

- ○ Density estimation
- ○ Groups or clusters in the data
- ○ Dimensionality reduction