



ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN  
VIETNAM - KOREA UNIVERSITY OF INFORMATION AND COMMUNICATION TECHNOLOGY

한-베정보통신기술대학교

Nhân bản – Phụng sự – Khai phóng

## Chapter 5

# Clustering Techniques

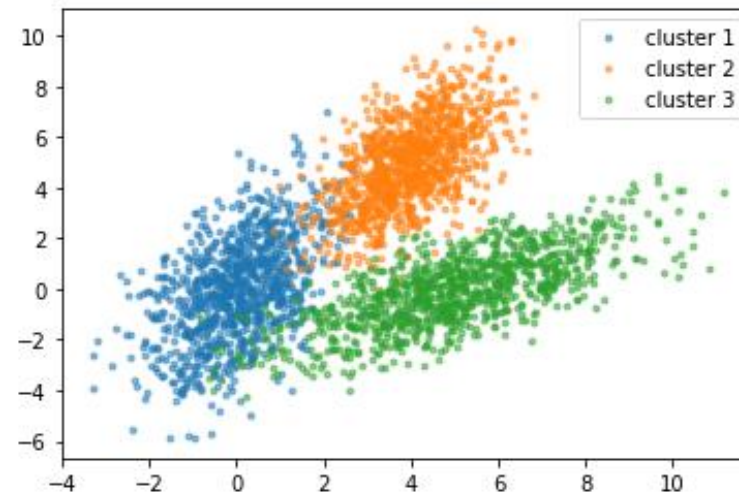
Machine Learning

- **Clustering Problems**
- **K-Means**
- **DBSCAN**
- **Gaussian Mixtures**

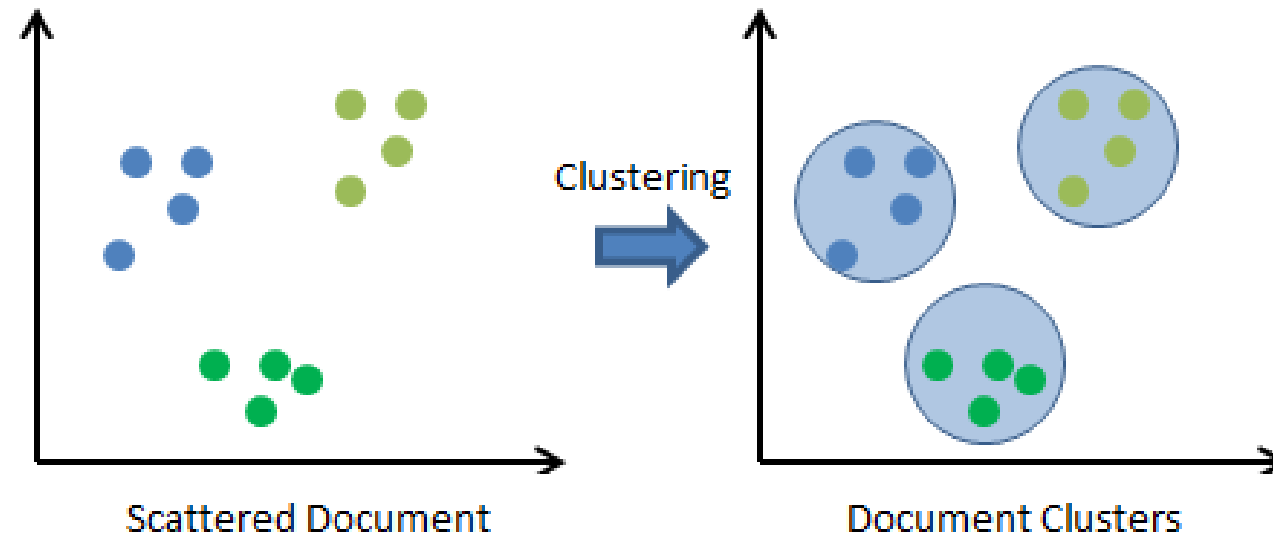
- **Clustering Problems**

- K-Means
- DBSCAN
- Gaussian Mixtures

- Unsupervised learning
- Sometimes the data form clusters, where examples within a cluster are similar to each other, and examples in different clusters are dissimilar:



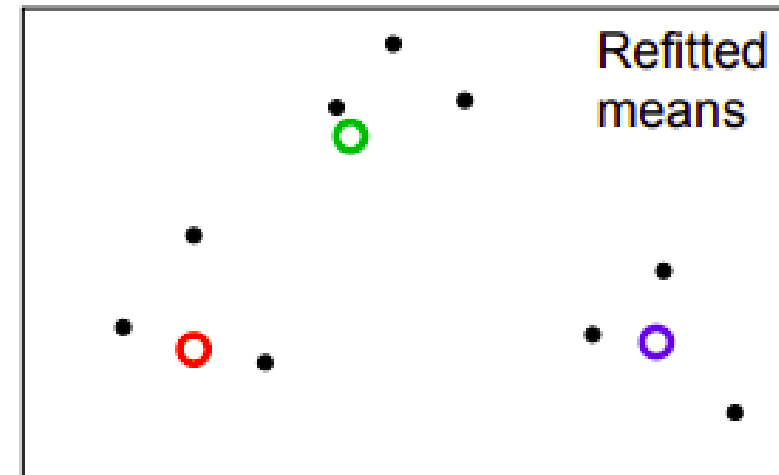
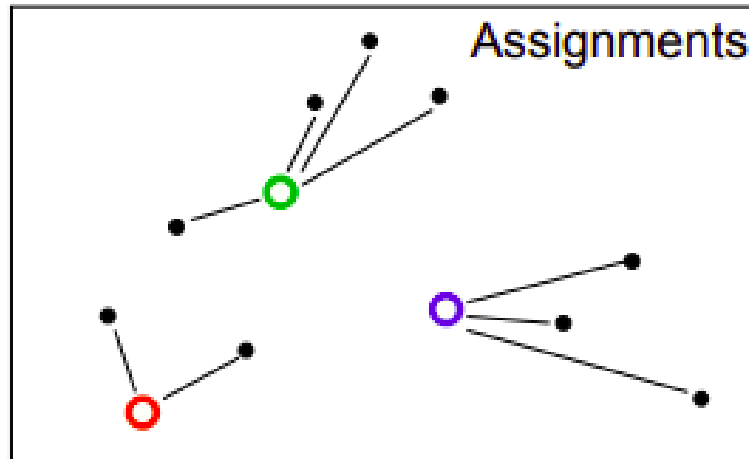
- Grouping data points into clusters, with no labels, is called **clustering**



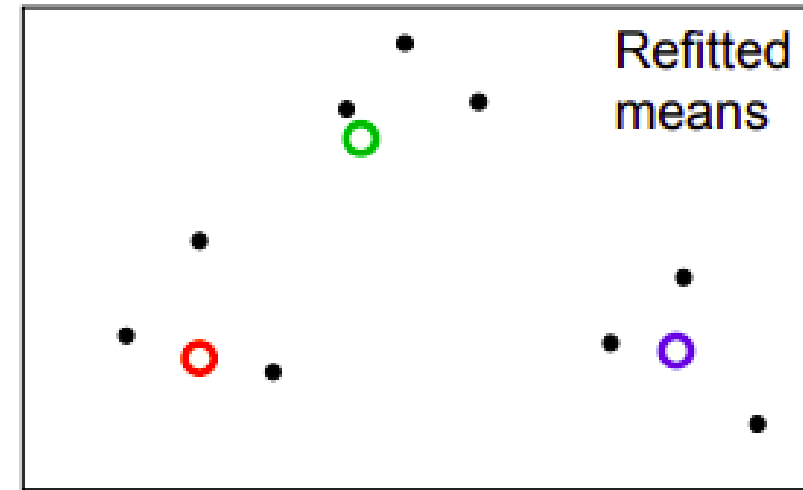
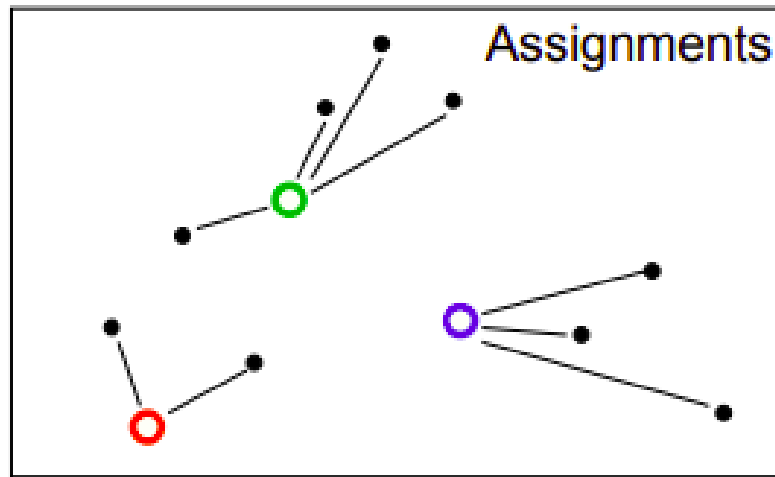
- Assume the data  $\{x(1), \dots, x(N)\}$  lives in a Euclidean space,  $x(n) \in \mathbb{R}^d$
- Assume the data belongs to  $K$  classes (patterns)
- Assume the data points from same class are similar, i.e. close in Euclidean distance.  
How can we identify those classes (data points that belong to each class)?

- Clustering Problems
- **K-Means**
- DBSCAN
- Gaussian Mixtures

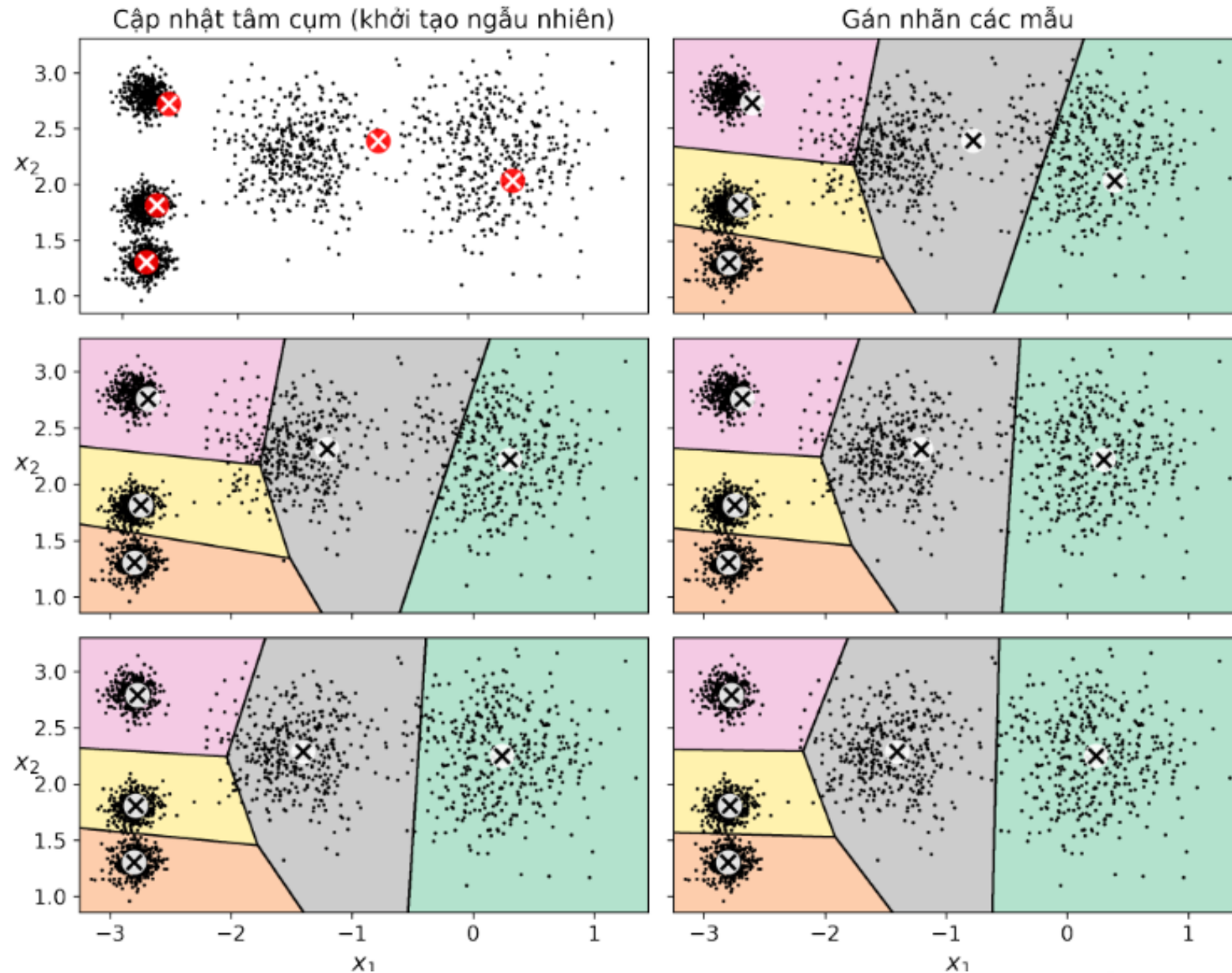
- Initialization: randomly initialize cluster centers
- The algorithm iteratively alternates between two steps:
  - Assignment step: Assign each data point to the closest cluster
  - Refitting step: Move each cluster center to the center of gravity of the data assigned to it

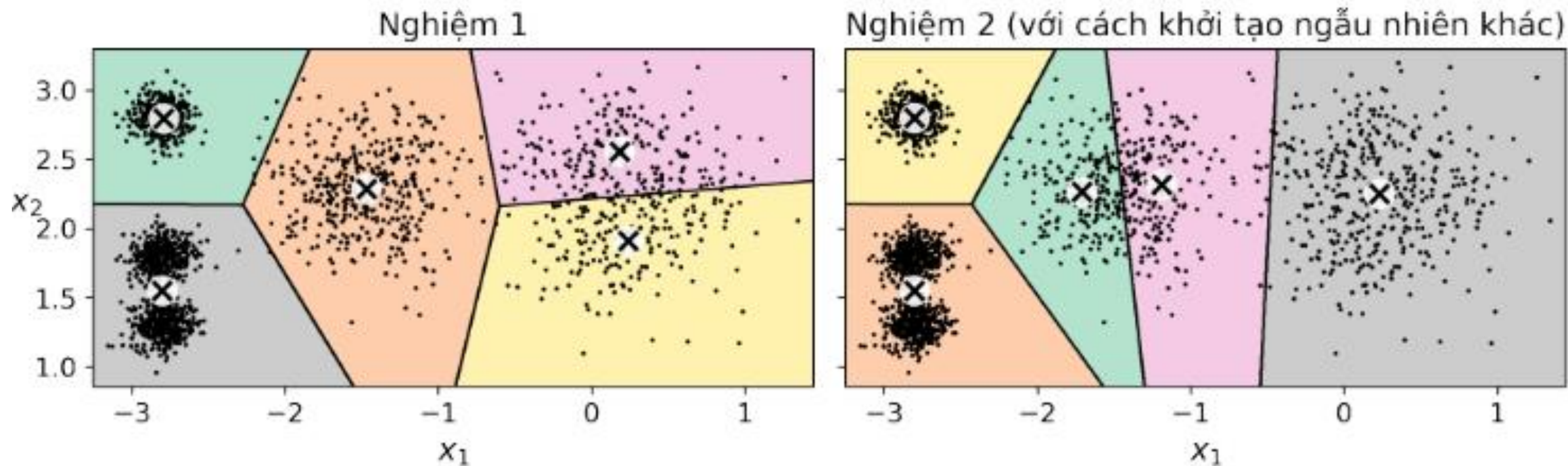


- K-means assumes there are K clusters, and each point is close to its cluster center (the mean of points in the cluster).
- If we knew the cluster assignment we could easily compute means.
- If we knew the means we could easily compute cluster assignment.
  - ⇒ Chicken and egg problem!
  - ⇒ Can show it is NP hard.
- Very simple (and useful) heuristic - start randomly and alternate between the two!



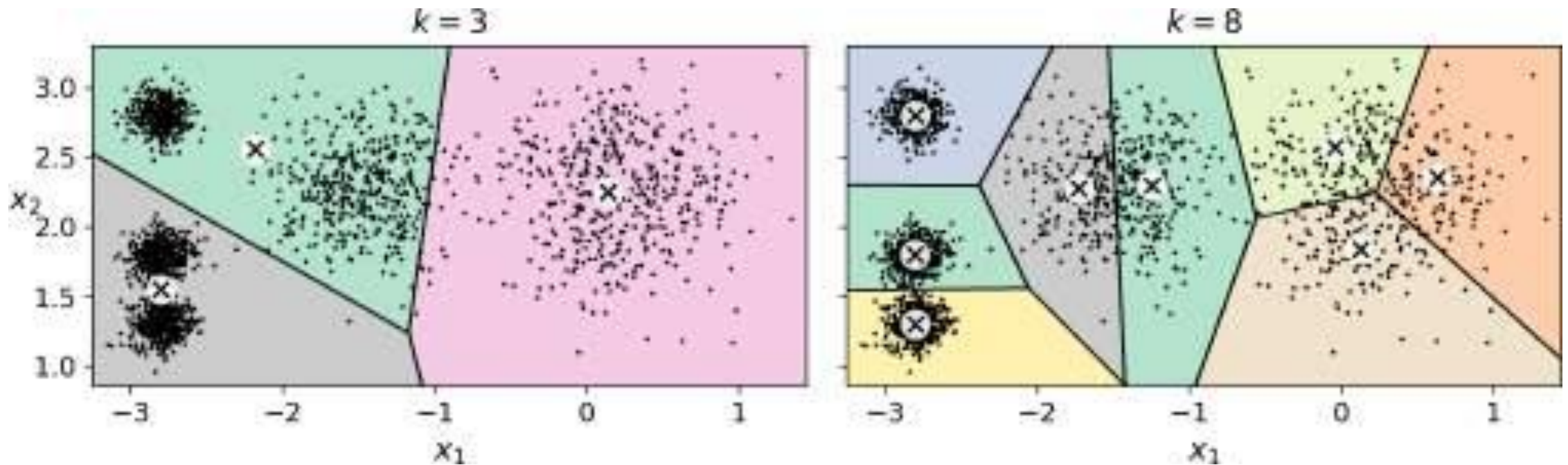






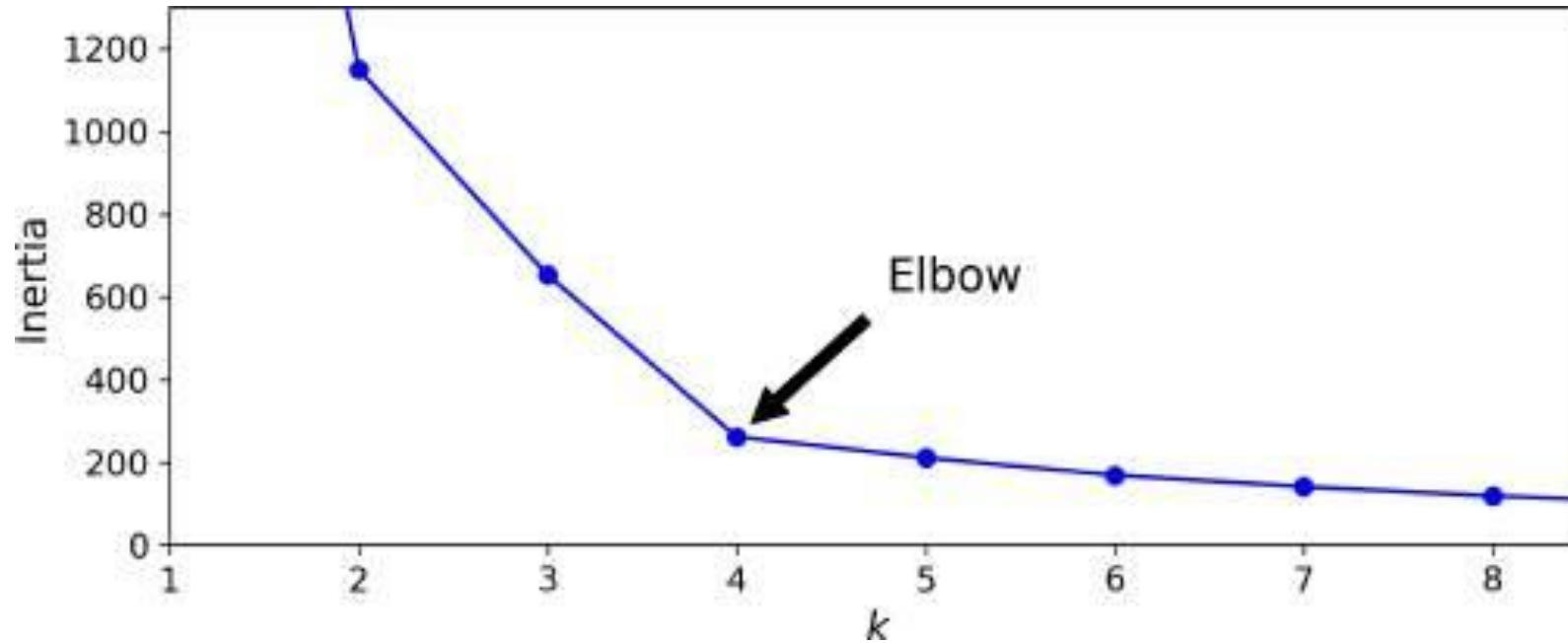
Hình 9.5. Các ngữ nghiệm chưa tối ưu trong trường hợp khởi tạo tâm cụm không tốt

- Finding the Optimal Number of Clusters



- Bad choices for the number of clusters

- Finding the Optimal Number of Clusters

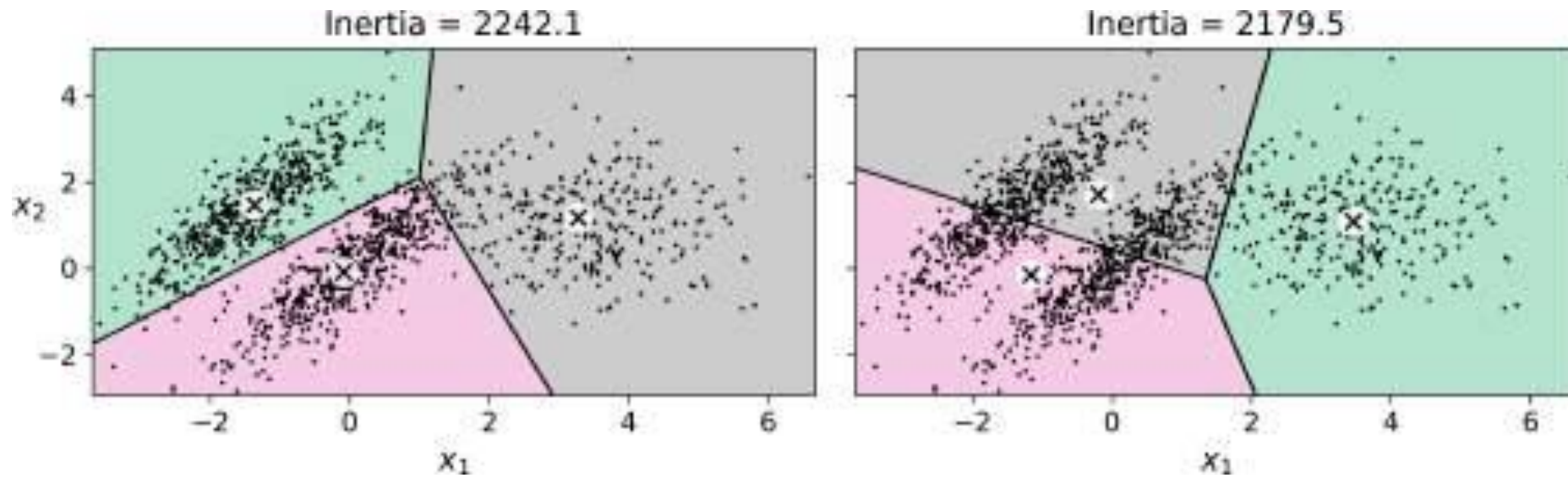


- Selecting the number of clusters  $k$  using the “elbow rule”



## • Limits of K-Means

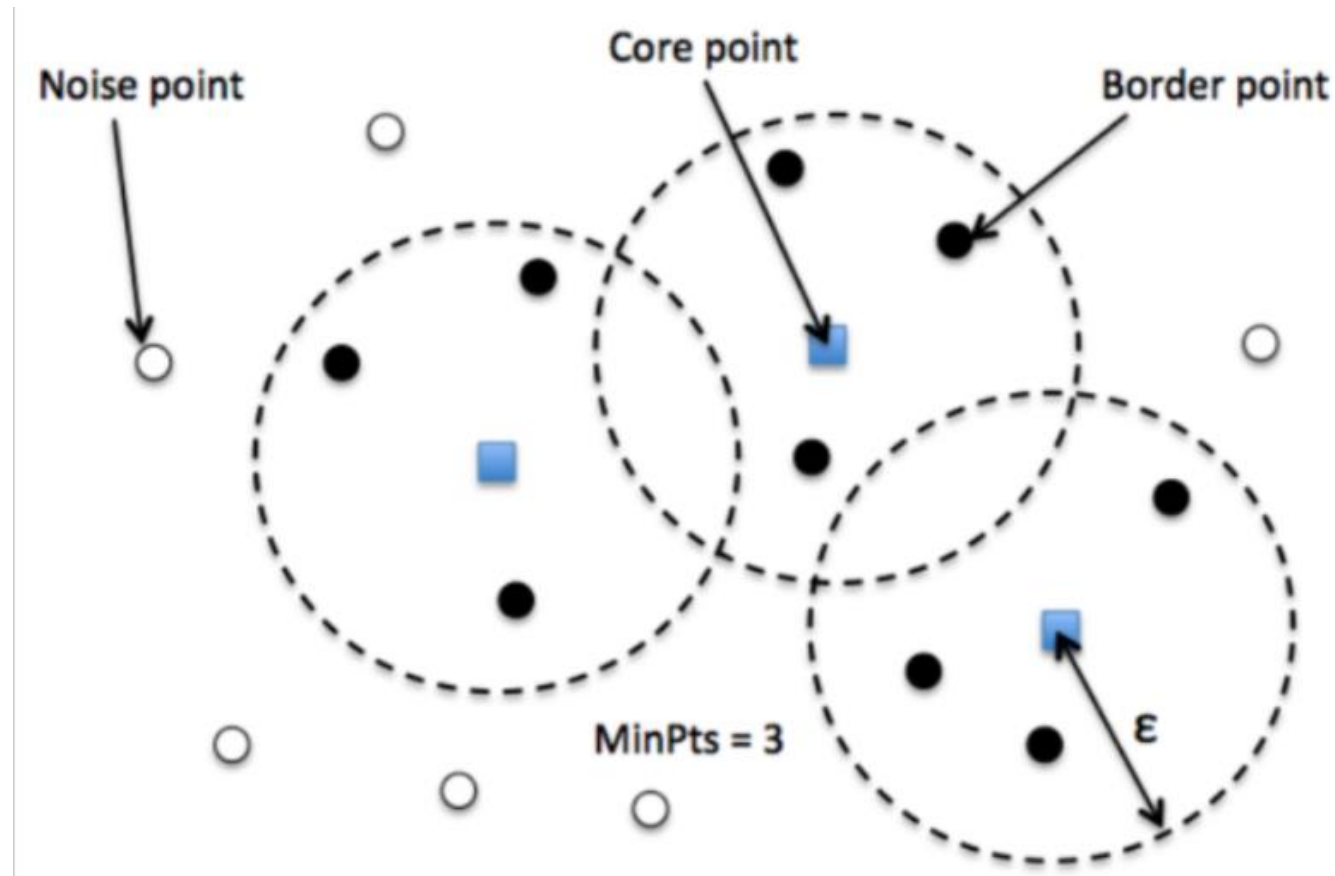
- K-Means does not behave very well when the clusters have varying sizes,
- different densities, or non-spherical shapes



K-Means fails to cluster these ellipsoidal blobs properly

- Clustering Problems
- K-Means
- **DBSCAN**
- Gaussian Mixtures

- DBSCAN – Density-Based Spatial Clustering of Applications with Noise
  - Core, Border, and Noise points

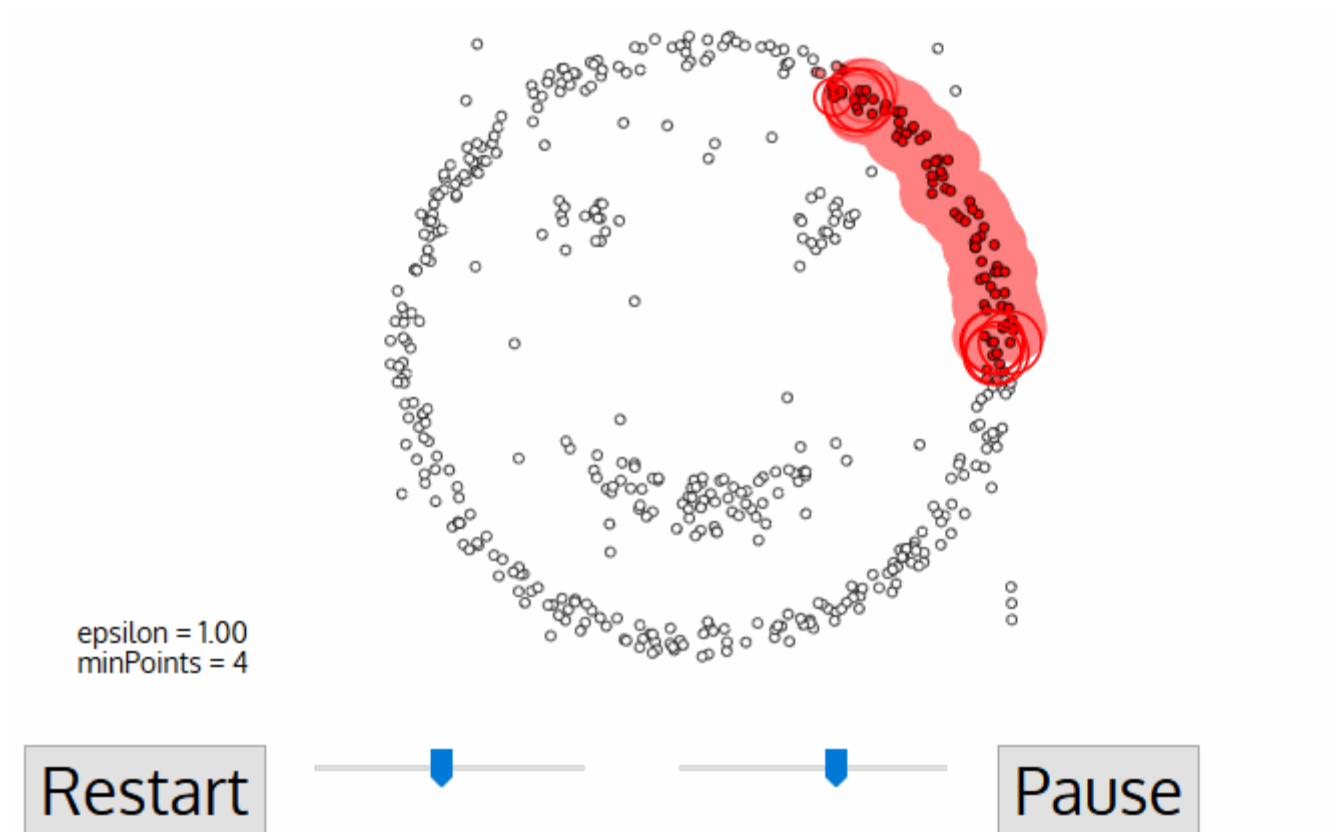


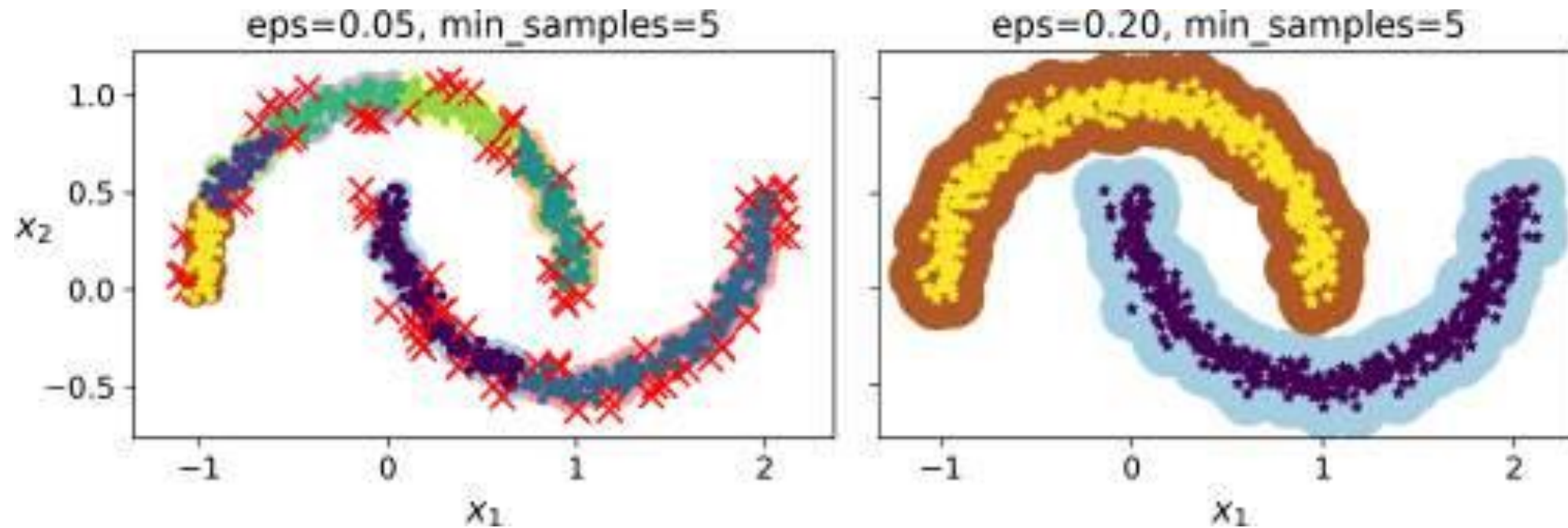
- Clusters as continuous regions of high density.
- DBSCAN algorithm:
  - For each instance, the algorithm counts how many instances are located within a small distance  $\epsilon$  (epsilon) from it. This region is called the instance's  $\epsilon$  neighborhood.
  - If an instance has at least `min_samples` instances in its  $\epsilon$ -neighborhood (including itself), then it is considered a core instance.
  - All instances in the neighborhood of a core instance belong to the same cluster.
  - Any instance that is not a core instance and does not have one in its neighborhood is considered an anomaly.



- Let ClusterCount=0. For every point  $p$ :
  - 1. If  $p$  it is not a core point, assign a null label to it [e.g., zero]
  - 2. If  $p$  is a core point, a new cluster is formed  
[with label ClusterCount:= ClusterCount+1]
- Then find all points density-reachable from  $p$  and classify them in the cluster. [Reassign the zero labels but not the others]
- Repeat this process until all of the points have been visited.  
(Since all the zero labels of border points have been reassigned in 2, the remaining points with zero label are noise).

- Time Complexity:  $O(n^2)$ —for each point it has to be determined if it is a core point, can be reduced to  $O(n \cdot \log(n))$  in lower dimensional spaces by using efficient data structures ( $n$  is the number of objects to be clustered);
- Space Complexity:  $O(n)$ .

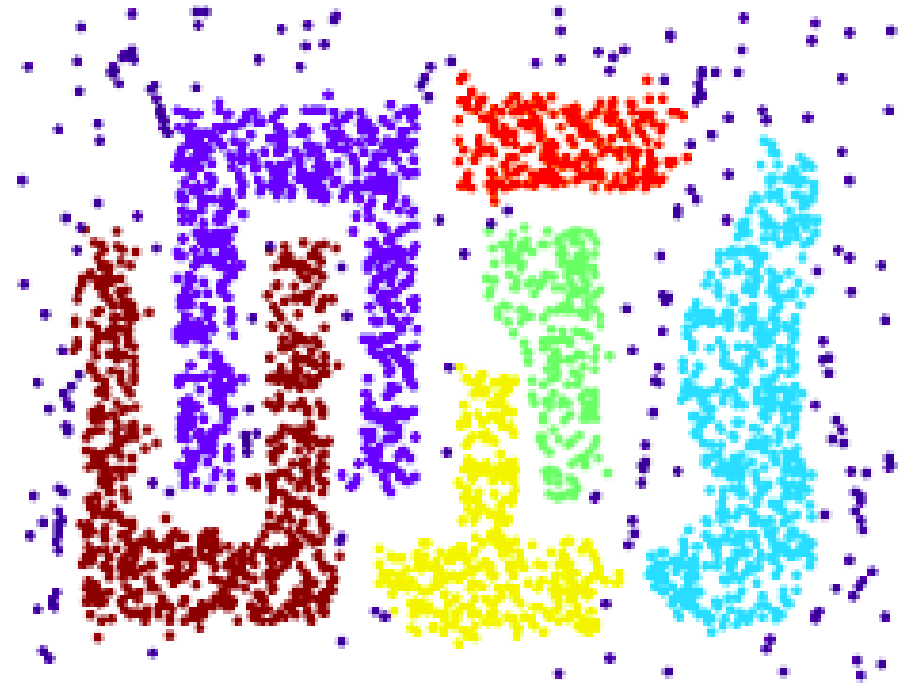




- DBSCAN clustering using two different neighborhood radiuses



Original Points



Clusters

- Clustering Problems
- K-Means
- DBSCAN
- **Gaussian Mixtures**



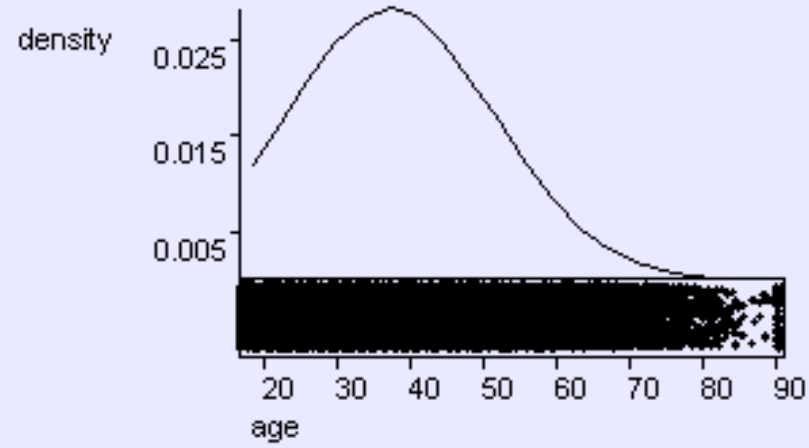
## Predicting wealth from age

wealth = poor

(prior = 0.760718)

1 mean cov

age 37.374 198.935

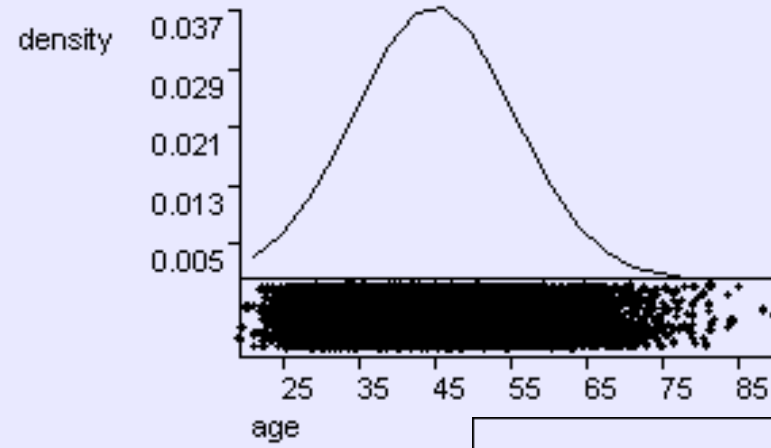


wealth = rich

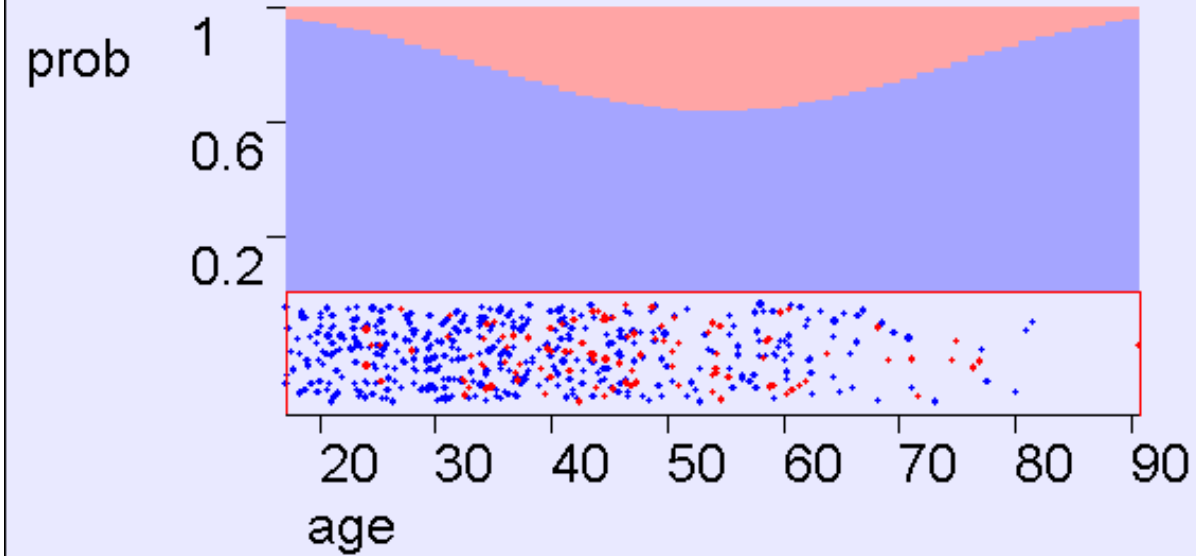
(prior = 0.239282)

1 mean cov

age 44.7727 111.618

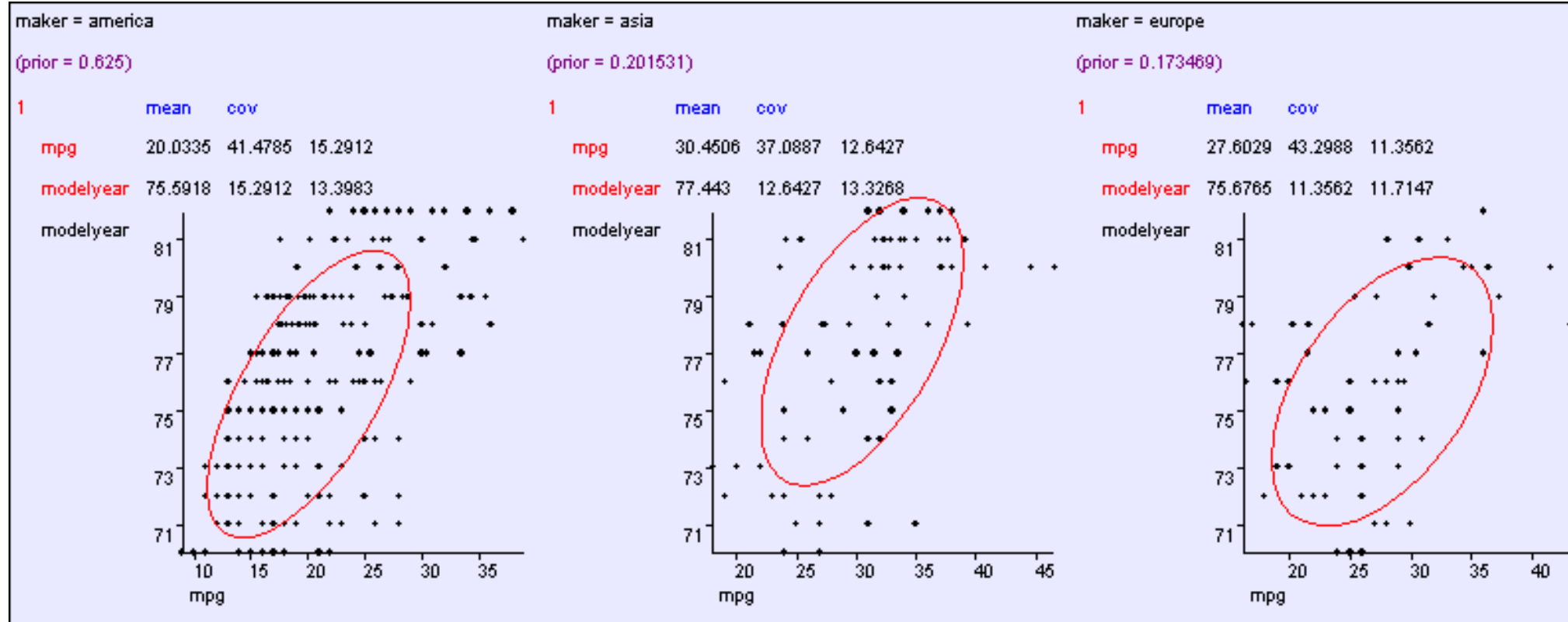


wealth values: poor rich



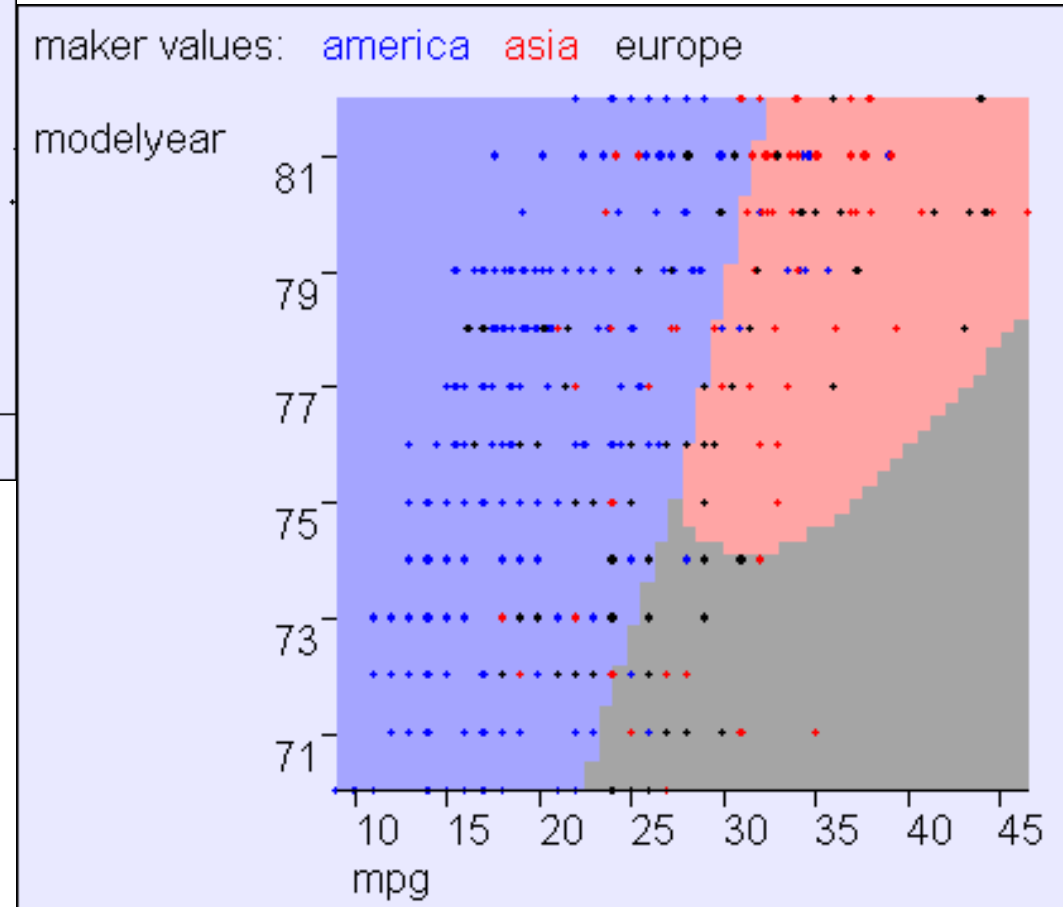
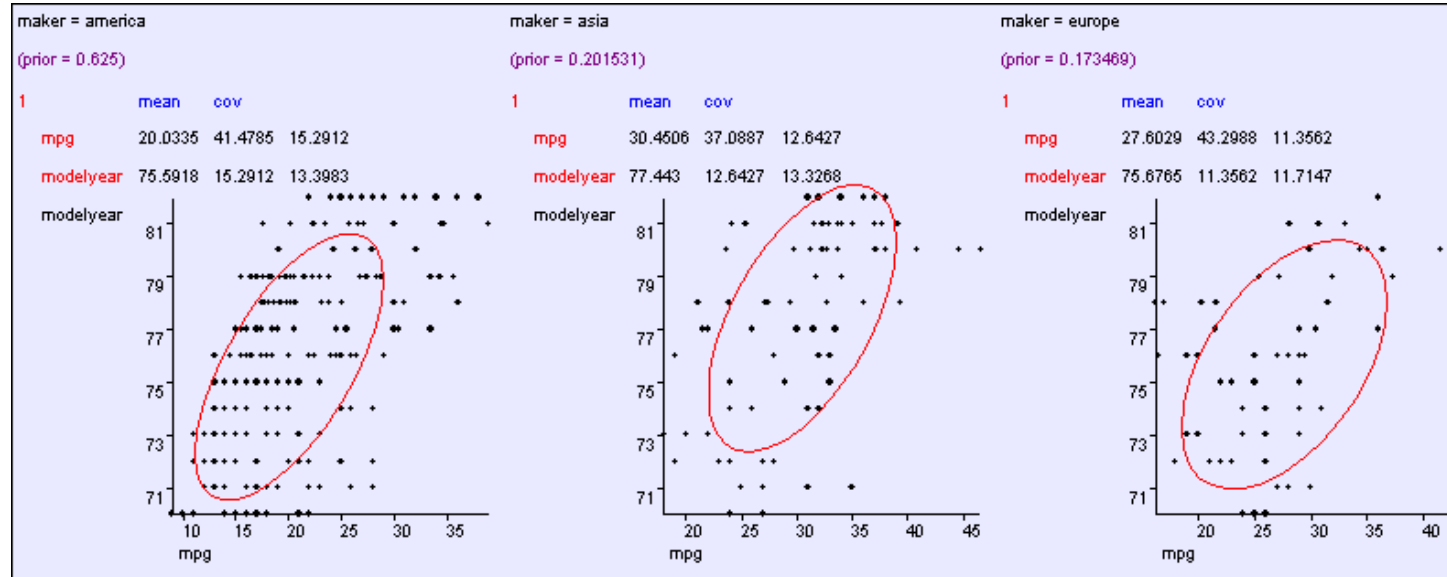


$$\Sigma = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12} & \cdots & \sigma_{1m} \\ \sigma_{12} & \sigma_{22}^2 & \cdots & \sigma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{1m} & \sigma_{2m} & \cdots & \sigma_{mm}^2 \end{pmatrix}$$

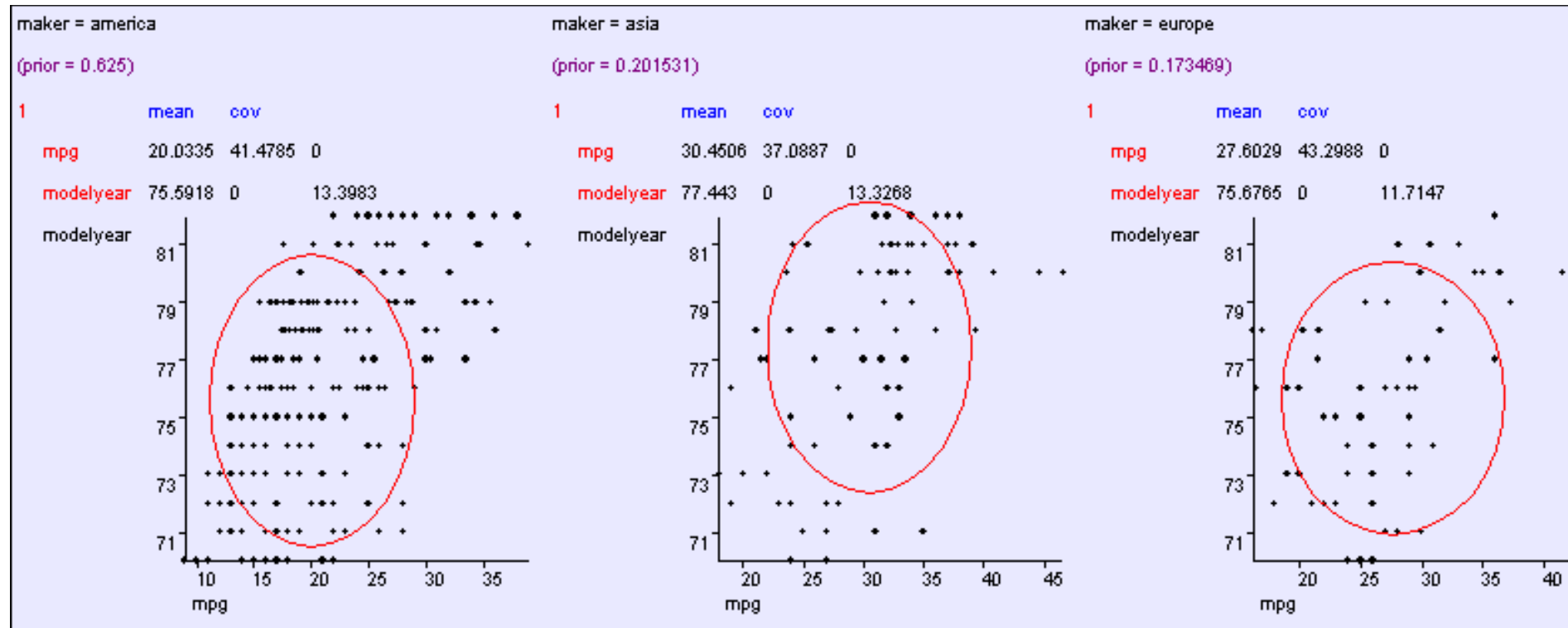


$$\Sigma = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12} & \cdots & \sigma_{1m} \\ \sigma_{12} & \sigma_{22}^2 & \cdots & \sigma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{1m} & \sigma_{2m} & \cdots & \sigma_{mm}^2 \end{pmatrix}$$

General:  $O(m^2)$  parameters

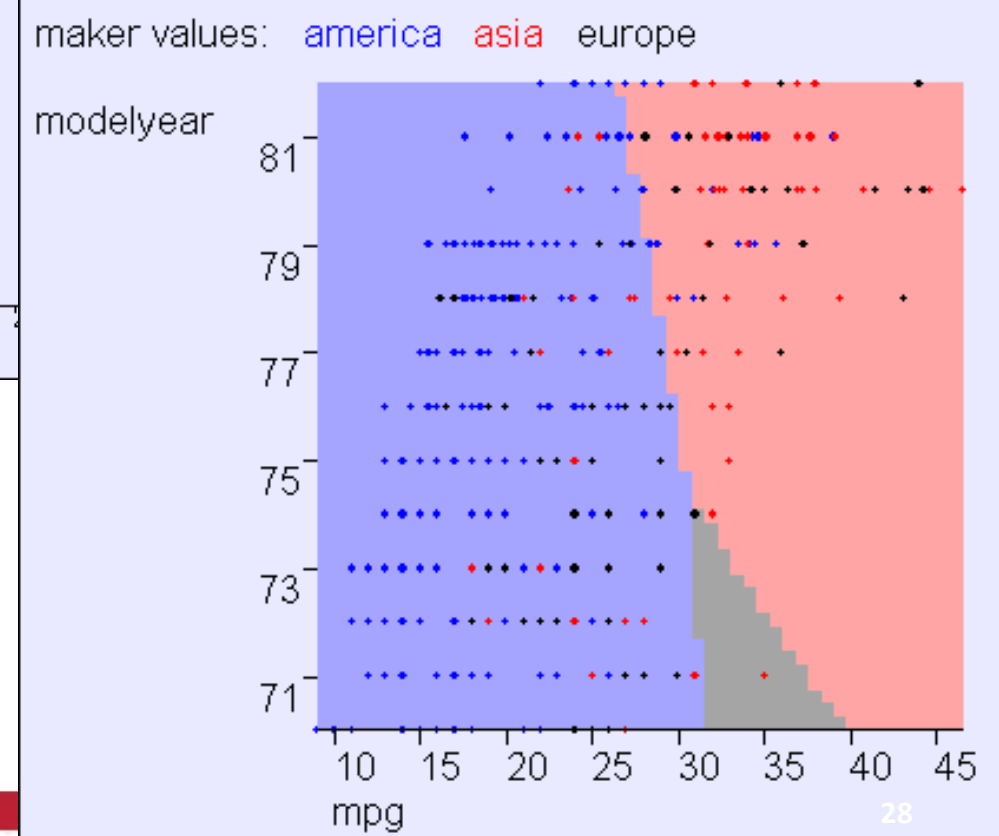
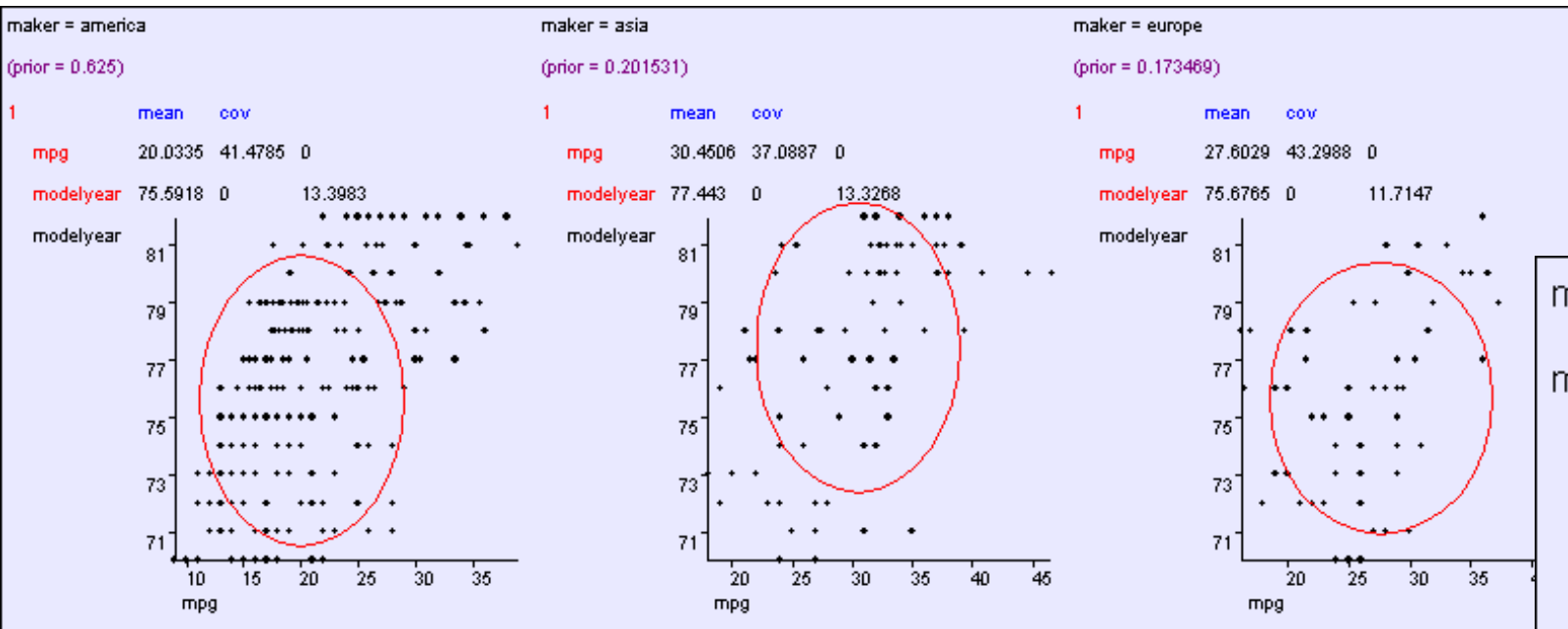


$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 & 0 \\ 0 & 0 & \sigma_3^2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_{m-1}^2 & 0 \\ 0 & 0 & 0 & \dots & 0 & \sigma_m^2 \end{pmatrix}$$

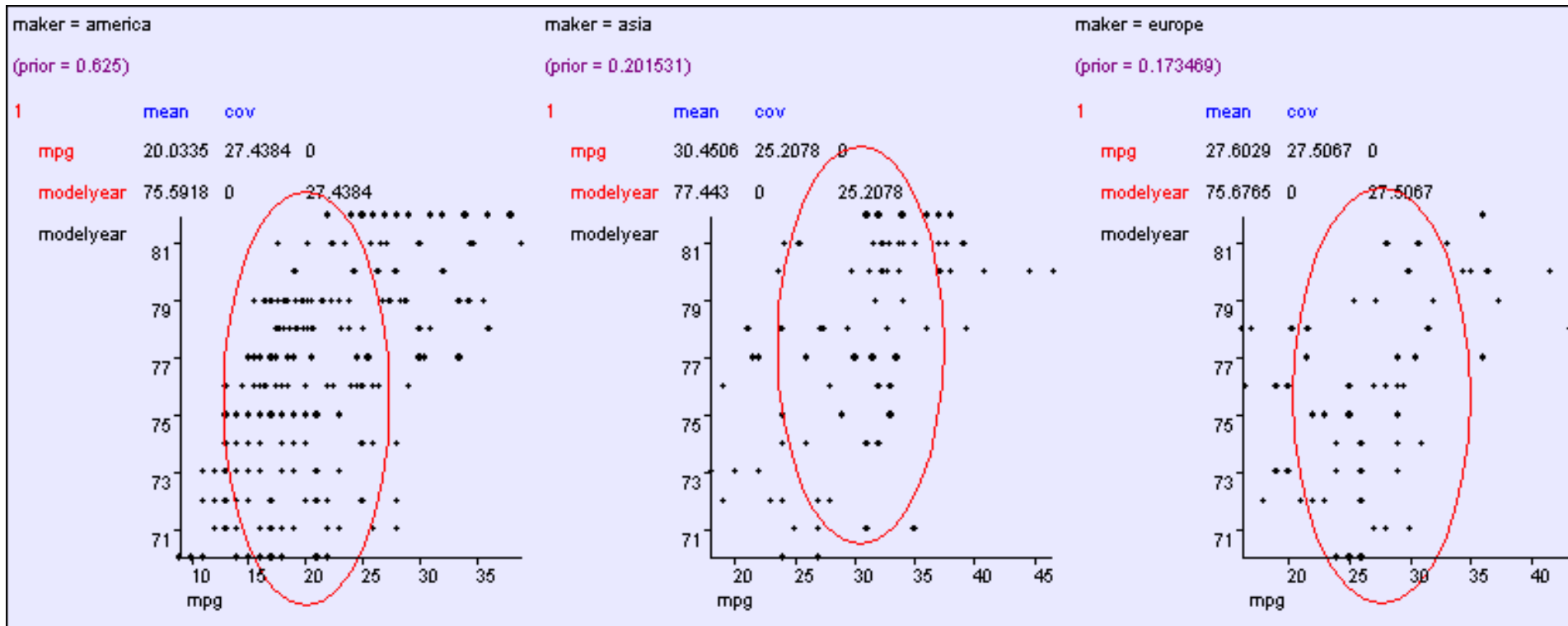


$$\Sigma = \begin{pmatrix} \sigma^2_1 & 0 & 0 & \dots & 0 & 0 \\ 0 & \sigma^2_2 & 0 & \dots & 0 & 0 \\ 0 & 0 & \sigma^2_3 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \sigma^2_{m-1} & 0 \\ 0 & 0 & 0 & \dots & 0 & \sigma^2_m \end{pmatrix}$$

# Aligned: $O(m)$ parameters

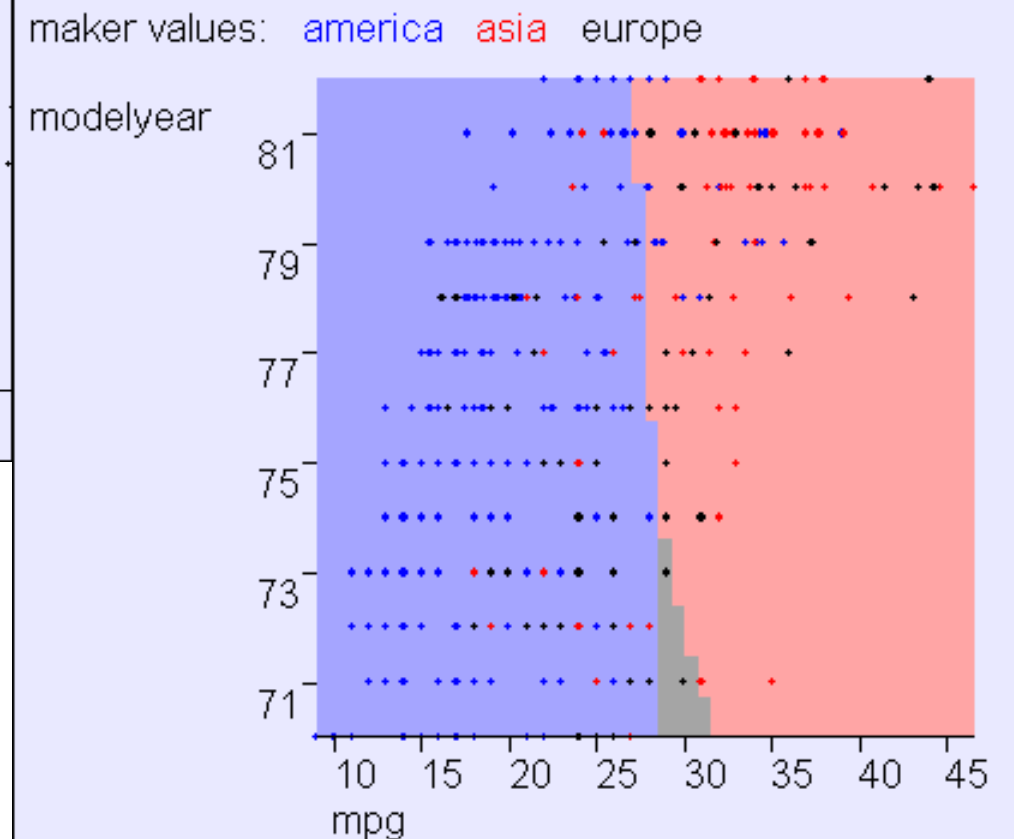
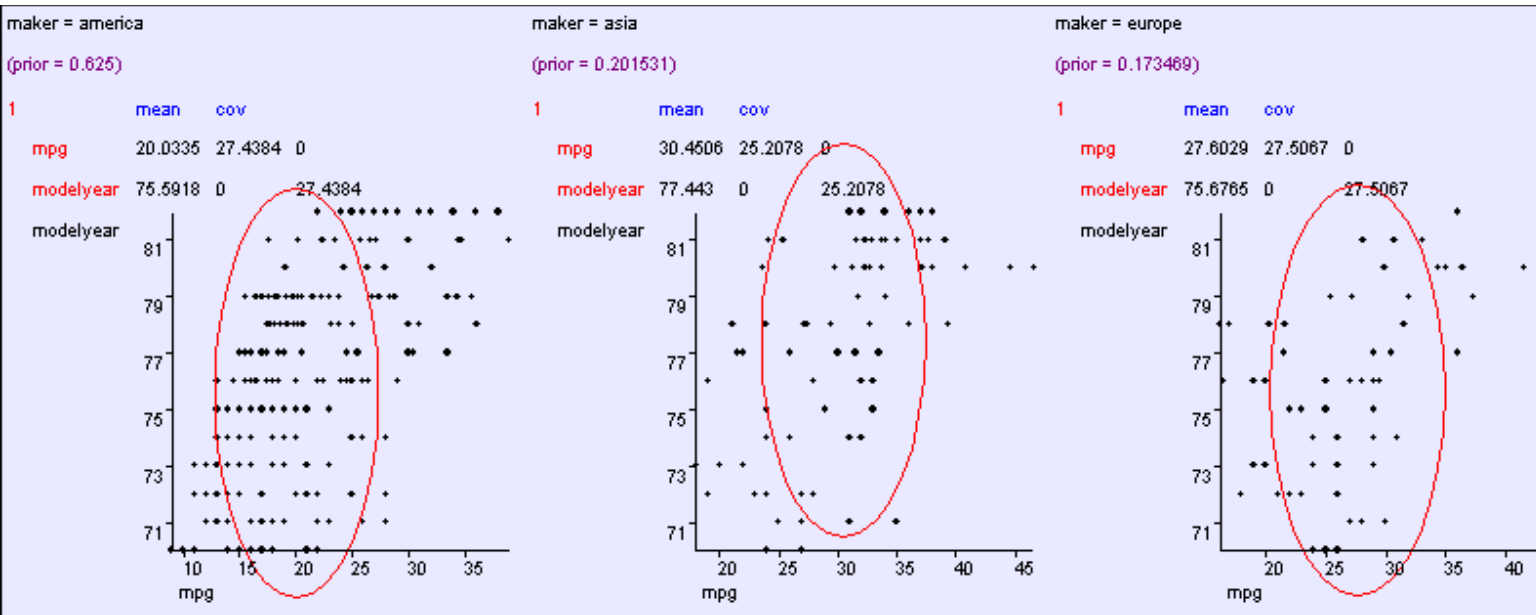


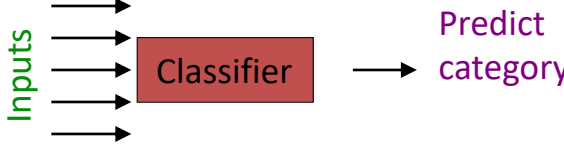

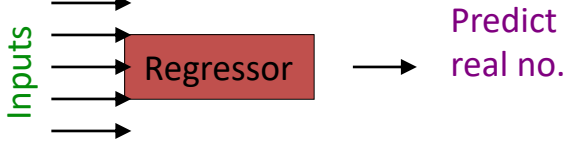
$$\Sigma = \begin{pmatrix} \sigma^2 & 0 & 0 & \dots & 0 & 0 \\ 0 & \sigma^2 & 0 & \dots & 0 & 0 \\ 0 & 0 & \sigma^2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \sigma^2 & 0 \\ 0 & 0 & 0 & \dots & 0 & \sigma^2 \end{pmatrix}$$



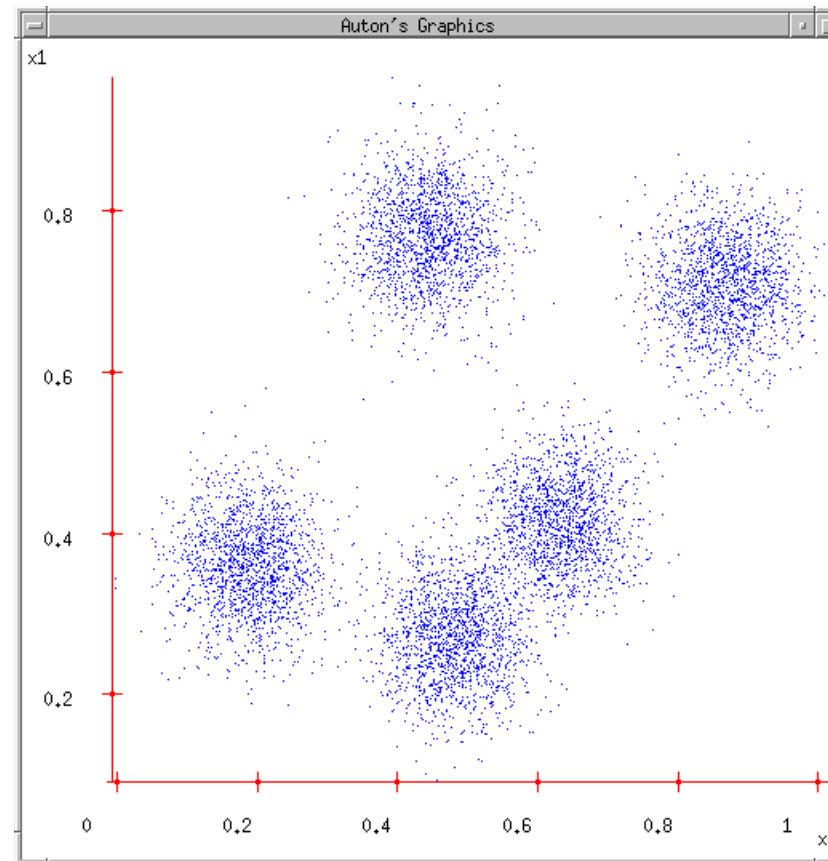
$$\Sigma = \begin{pmatrix} \sigma^2 & 0 & 0 & \dots & 0 & 0 \\ 0 & \sigma^2 & 0 & \dots & 0 & 0 \\ 0 & 0 & \sigma^2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \sigma^2 & 0 \\ 0 & 0 & 0 & \dots & 0 & \sigma^2 \end{pmatrix}$$

# Spherical: $O(1)$ cov parameters



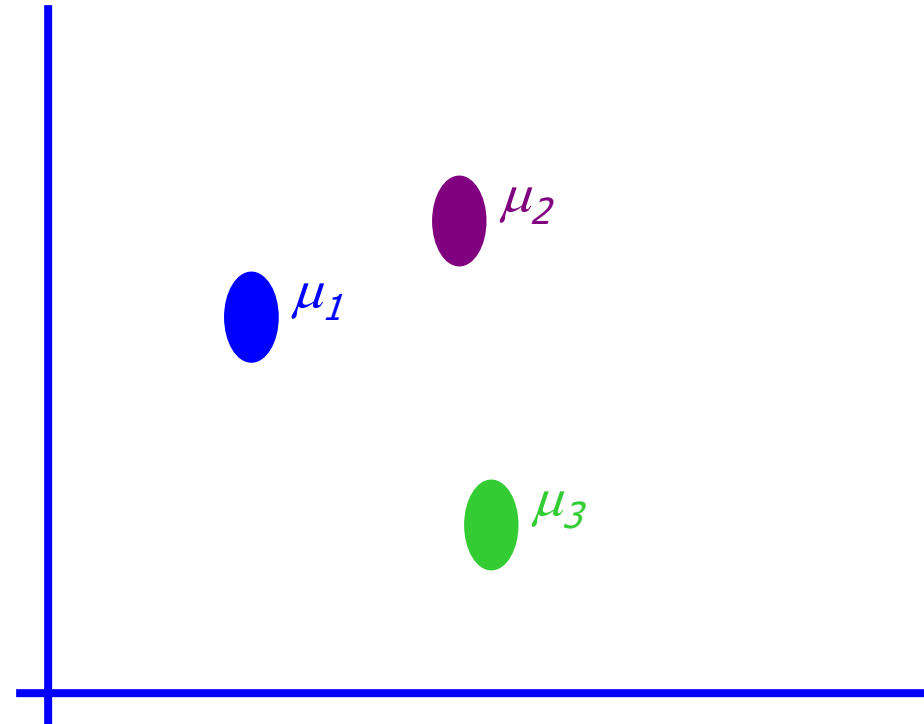
	Categorical inputs only	Real-valued inputs only	Mixed Real / Cat okay
	Joint BC Naïve BC	Gauss BC	Dec Tree
	Joint DE Naïve DE	Gauss DE	
			

What if we want to do density estimation with multimodal or clumpy data?



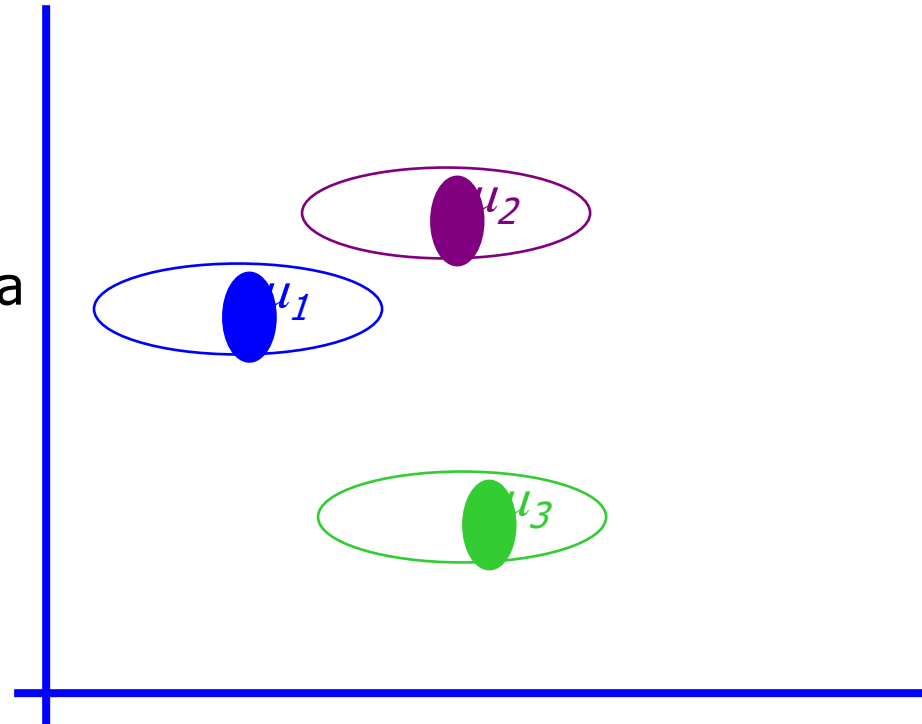


- There are  $k$  components. The  $i$ 'th component is called  $\omega_i$
- Component  $\omega_i$  has an associated mean vector  $\mu_i$



- There are  $k$  components. The  $i$ 'th component is called  $\omega_i$
- Component  $\omega_i$  has an associated mean vector  $\mu_i$
- Each component generates data from a Gaussian with mean  $\mu_i$  and covariance matrix  $\sigma^2 \mathbf{I}$

Assume that each datapoint is generated according to the following recipe:



- There are  $k$  components. The  $i$ 'th component is called  $\omega_i$
- Component  $\omega_i$  has an associated mean vector  $\mu_i$
- Each component generates data from a Gaussian with mean  $\mu_i$  and covariance matrix  $\sigma^2 \mathbf{I}$

Assume that each datapoint is generated according to the following recipe:

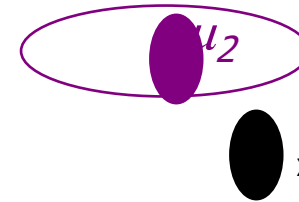
1. Pick a component at random. Choose component  $i$  with probability  $P(\omega_i)$ .



- There are  $k$  components. The  $i$ 'th component is called  $\omega_i$
- Component  $\omega_i$  has an associated mean vector  $\mu_i$
- Each component generates data from a Gaussian with mean  $\mu_i$  and covariance matrix  $\sigma^2 \mathbf{I}$

Assume that each datapoint is generated according to the following recipe:

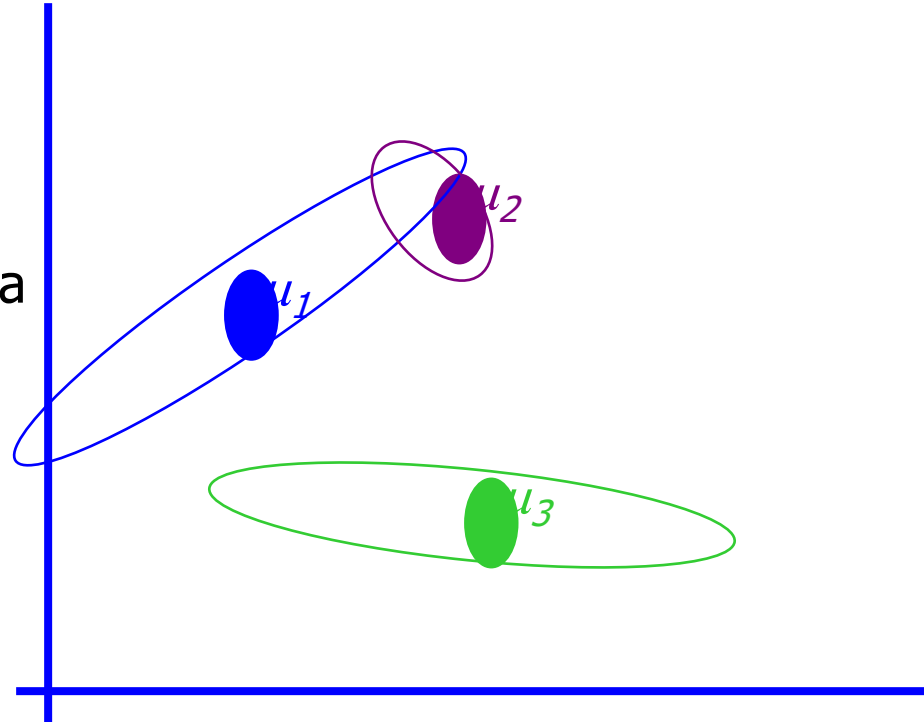
1. Pick a component at random. Choose component  $i$  with probability  $P(\omega_i)$ .
2. Datapoint  $\sim N(\mu_i, \sigma^2 \mathbf{I})$

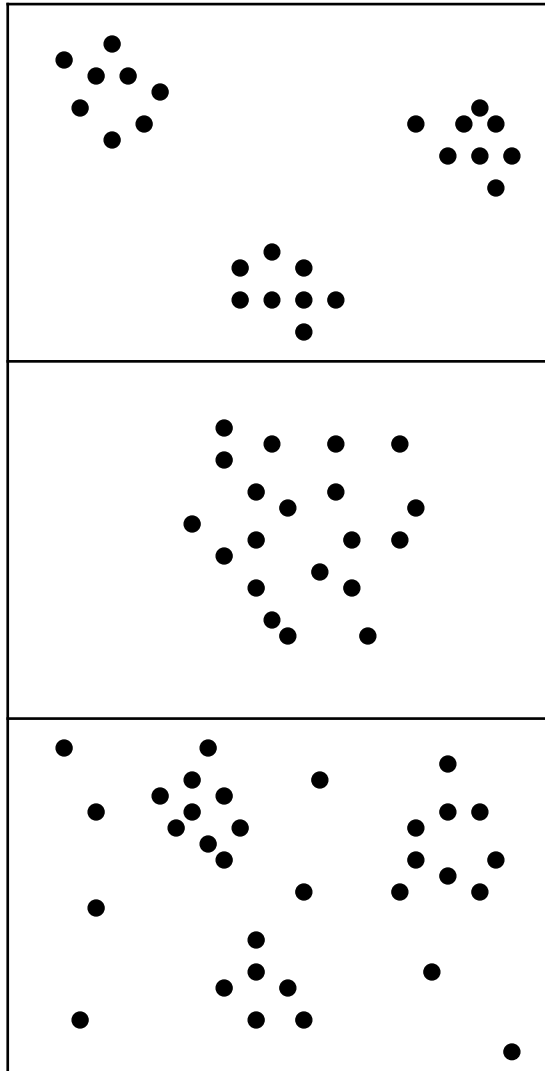


- There are  $k$  components. The  $i$ 'th component is called  $\omega_i$
- Component  $\omega_i$  has an associated mean vector  $\mu_i$
- Each component generates data from a Gaussian with mean  $\mu_i$  and covariance matrix  $\Sigma_i$

Assume that each datapoint is generated according to the following recipe:

1. Pick a component at random. Choose component  $i$  with probability  $P(\omega_i)$ .
2. Datapoint  $\sim N(\mu_i, \Sigma_i)$





Sometimes easy

Sometimes impossible

and sometimes  
in between

IN CASE YOU'RE WONDERING WHAT THESE DIAGRAMS ARE, THEY SHOW 2-d UNLABELED DATA ( $\mathbf{x}$  VECTORS) DISTRIBUTED IN 2-d SPACE. THE TOP ONE HAS THREE VERY CLEAR GAUSSIAN CENTERS

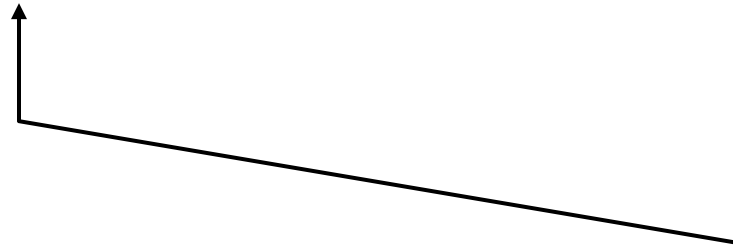
We have  $x_1, x_2, \dots, x_N$

We know  $P(w_1) P(w_2) \dots P(w_k)$

We know  $\sigma$

$P(x | w_i, \mu_1, \dots, \mu_k) = \text{Prob that an observation from class } w_i \text{ would have value } x \text{ given class means } \mu_1 \dots \mu_k$

Can we write an expression for that?



We have  $x_1 x_2 \dots x_n$

We have  $P(w_1) \dots P(w_k)$ . We have  $\sigma$ .

We can define, for any  $x$ ,  $P(x|w_i, \mu_1, \mu_2 \dots \mu_k)$

Can we define  $P(x \mid \mu_1, \mu_2 \dots \mu_k)$  ?

Can we define  $P(x_1, x_1, \dots x_n \mid \mu_1, \mu_2 \dots \mu_k)$  ?

[YES, IF WE ASSUME THE  $x_i$ 'S WERE DRAWN INDEPENDENTLY]



We now have a procedure s.t. if you give me a guess at  $\mu_1, \mu_2 \dots \mu_k$ ,  
I can tell you the prob of the unlabeled data given those  $\mu$ 's.

Suppose  $x$ 's are 1-dimensional.

(From Duda and Hart)

There are two classes;  $w_1$  and  $w_2$

$$P(w_1) = 1/3 \quad P(w_2) = 2/3 \quad \sigma = 1.$$

There are 25 unlabeled datapoints

$$x_1 = 0.608$$

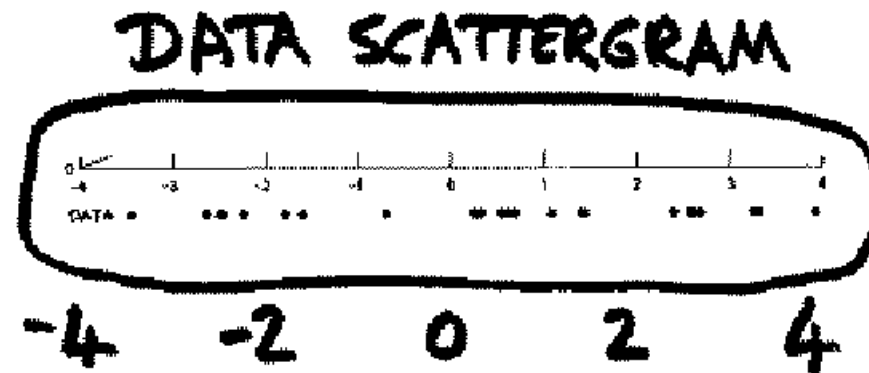
$$x_2 = -1.590$$

$$x_3 = 0.235$$

$$x_4 = 3.949$$

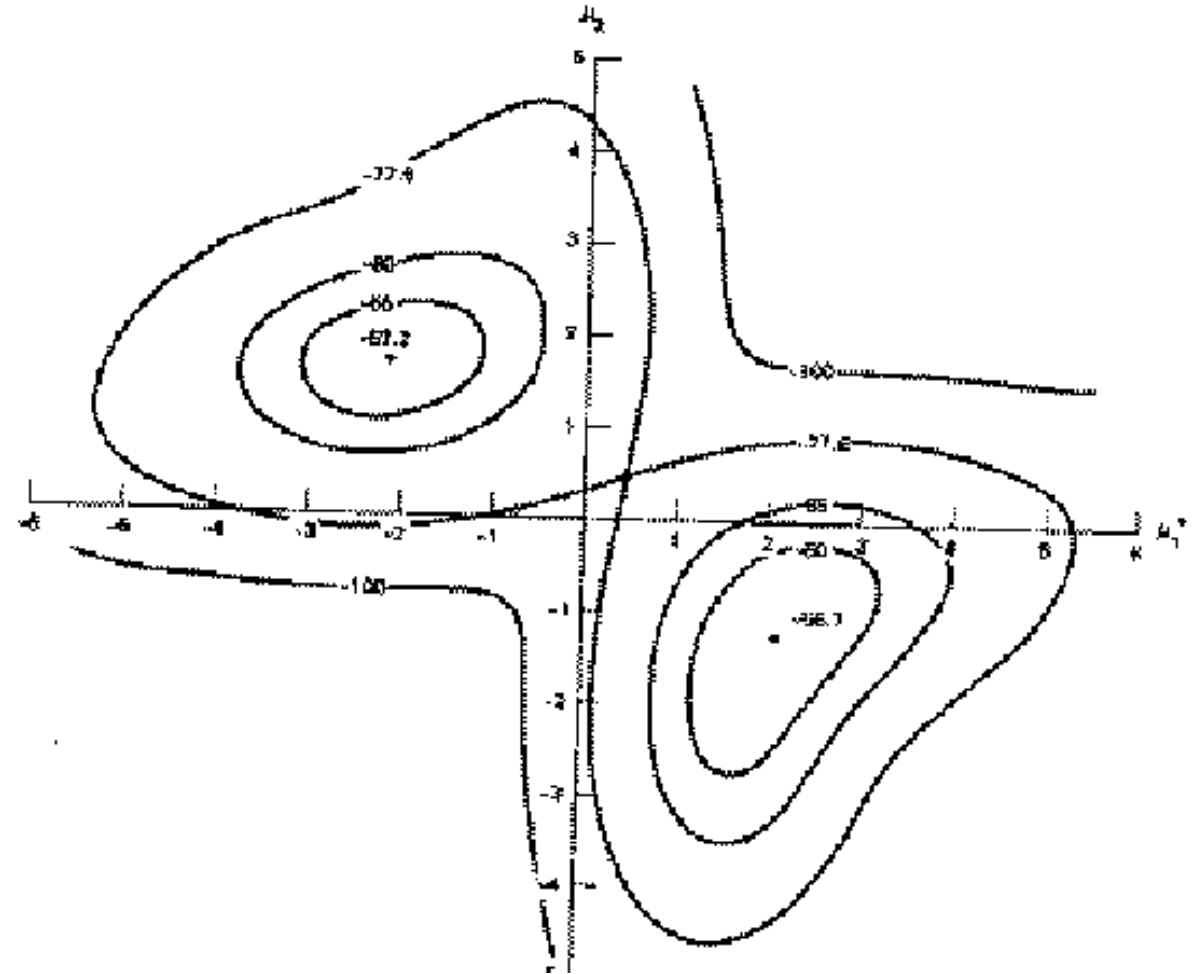
:

$$x_{25} = -0.712$$



Graph of

$\log P(x_1, x_2 \dots x_{25} \mid \mu_1, \mu_2)$   
against  $\mu_1 (\rightarrow)$  and  $\mu_2 (\uparrow)$



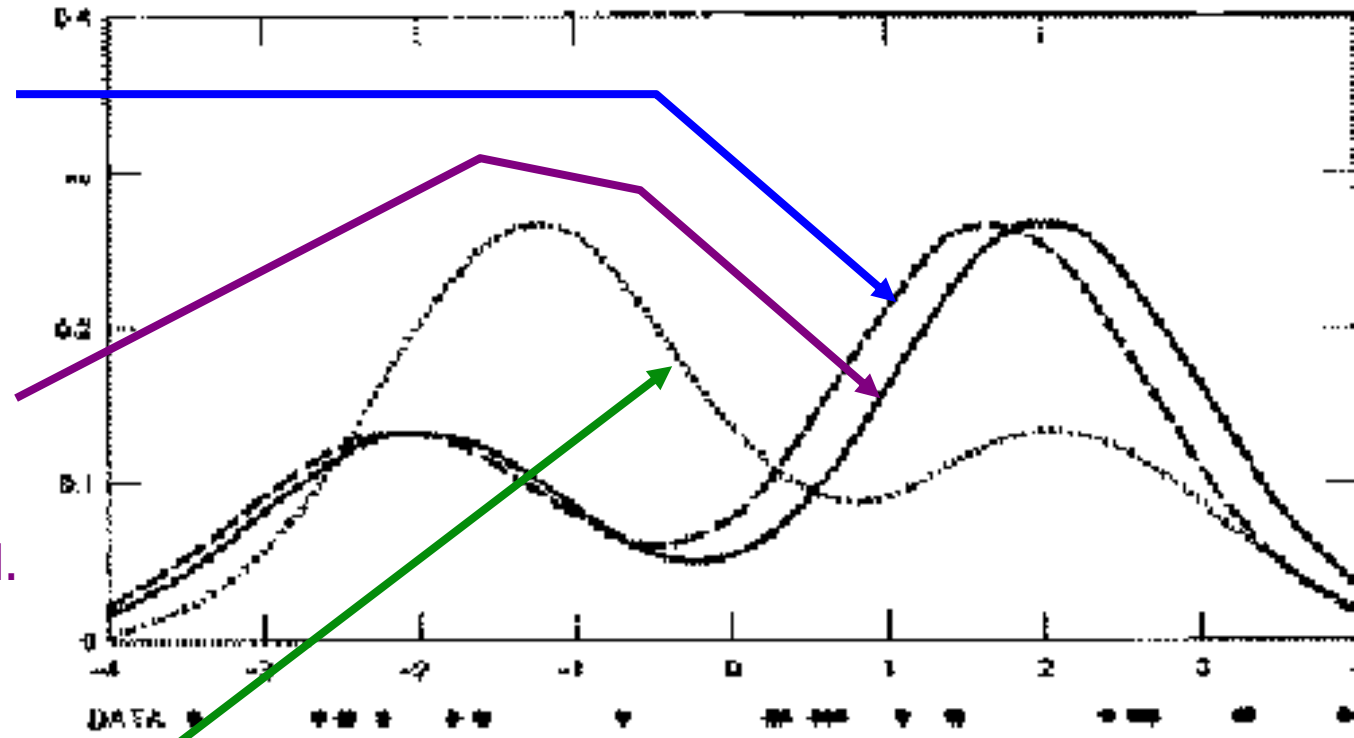
Max likelihood =  $(\mu_1 = -2.13, \mu_2 = 1.668)$

Local minimum, but very close to global at  $(\mu_1 = 2.085, \mu_2 = -1.257)^*$

\* corresponds to switching  $w_1 + w_2$ .

We can graph the prob. dist. function of data given our  $\mu_1$  and  $\mu_2$  estimates.

We can also graph the true function from which the data was randomly generated.



- They are close. Good.
- The 2<sup>nd</sup> solution tries to put the "2/3" hump where the "1/3" hump should go, and vice versa.
- In this example unsupervised is almost as good as supervised. If the  $x_1 .. x_{25}$  are given the class which was used to learn them, then the results are ( $\mu_1 = -2.176$ ,  $\mu_2 = 1.684$ ). Unsupervised got ( $\mu_1 = -2.13$ ,  $\mu_2 = 1.668$ ).

We can compute  $P(\text{data} \mid \mu_1, \mu_2 \dots \mu_k)$

How do we find the  $\mu_i$ 's which give max. likelihood?

- The normal max likelihood trick:

Set  $\frac{\partial}{\partial \mu_i} \log \text{Prob} (\dots) = 0$

$\mu_i$

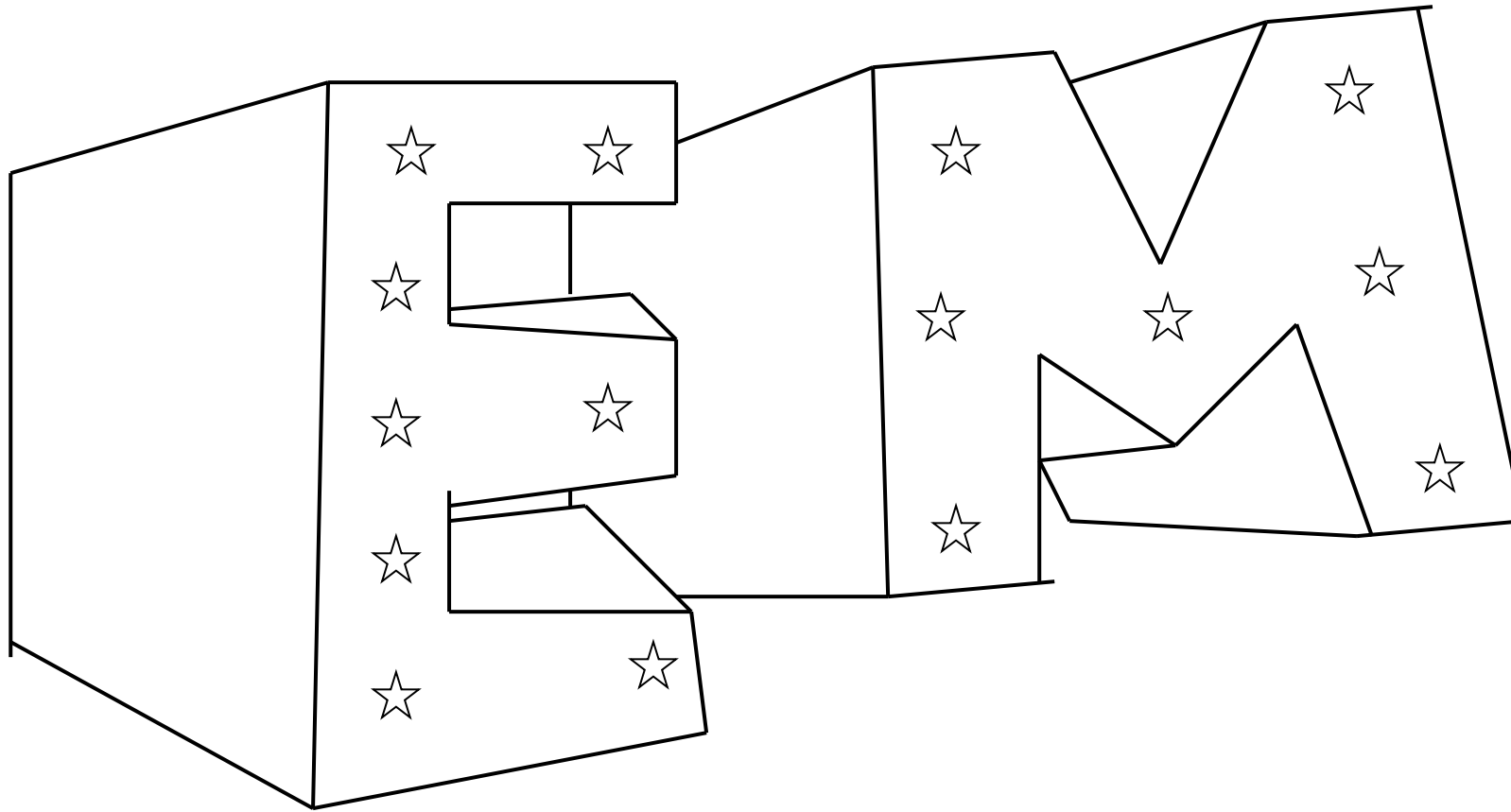
and solve for  $\mu_i$ 's.

# Here you get non-linear non-analytically- solvable equations

- Use gradient descent

Slow but doable

- Use a much faster, cuter, and recently very popular method...



- We'll get back to unsupervised learning soon.
- But now we'll look at an even simpler case with hidden information.
- The EM algorithm
  - ❑ Can do trivial things, such as the contents of the next few slides.
  - ❑ An excellent way of doing our unsupervised learning problem, as we'll see.
  - ❑ Many, many other uses, including inference of Hidden Markov Models (future lecture).

Let events be “grades in a class”

$w_1$  = Gets an A

$$P(A) = \frac{1}{2}$$

$w_2$  = Gets a B

$$P(B) = \mu$$

$w_3$  = Gets a C

$$P(C) = 2\mu$$

$w_4$  = Gets a D

$$P(D) = \frac{1}{2} - 3\mu$$

(Note  $0 \leq \mu \leq 1/6$ )

Assume we want to estimate  $\mu$  from data. In a given class there were

a A's

b B's

c C's

d D's

What's the maximum likelihood estimate of  $\mu$  given a,b,c,d ?

Let events be “grades in a class”

$w_1$  = Gets an A

$w_2$  = Gets a B

$w_3$  = Gets a C

$w_4$  = Gets a D

$P(A) = \frac{1}{2}$

$P(B) = \mu$

$P(C) = 2\mu$

$P(D) = \frac{1}{2} - 3\mu$

(Note  $0 \leq \mu \leq 1/6$ )

Assume we want to estimate  $\mu$  from data. In a given class there were

a A's

b B's

c C's

d D's

What's the maximum likelihood estimate of  $\mu$  given a,b,c,d ?



$$P(A) = \frac{1}{2} \quad P(B) = \mu \quad P(C) = 2\mu \quad P(D) = \frac{1}{2} - 3\mu$$

$$P(a, b, c, d \mid \mu) = K \left(\frac{1}{2}\right)^a (\mu)^b (2\mu)^c \left(\frac{1}{2} - 3\mu\right)^d$$

$$\log P(a, b, c, d \mid \mu) = \log K + a \log \frac{1}{2} + b \log \mu + c \log 2\mu + d \log \left(\frac{1}{2} - 3\mu\right)$$

$$\text{FOR MAX LIKE } \mu, \text{ SET } \frac{\partial \text{LogP}}{\partial \mu} = 0$$

$$\frac{\partial \text{LogP}}{\partial \mu} = \frac{b}{\mu} + \frac{2c}{2\mu} - \frac{3d}{1/2 - 3\mu} = 0$$

$$\text{Gives max like } \mu = \frac{b + c}{6(b + c + d)}$$

So if class got

$$\text{Max like } \mu = \frac{1}{10}$$

**Boring, but true!**

A	B	C	D
14	6	9	10

Someone tells us that

Number of High grades (A's + B's) =  $h$

Number of C's =  $c$

Number of D's =  $d$

What is the max. like estimate of  $\mu$  now?

REMEMBER

$$P(A) = \frac{1}{2}$$

$$P(B) = \mu$$

$$P(C) = 2\mu$$

$$P(D) = \frac{1}{2} - 3\mu$$

Someone tells us that

Number of High grades (A's + B's) =  $h$

Number of C's =  $c$

Number of D's =  $d$

REMEMBER

$$P(A) = \frac{1}{2}$$

$$P(B) = \mu$$

$$P(C) = 2\mu$$

$$P(D) = \frac{1}{2} - 3\mu$$

What is the max. like estimate of  $\mu$  now?

We can answer this question circularly:

## EXPECTATION

If we know the value of  $\mu$  we could compute the expected value of  $a$  and  $b$

Since the ratio  $a:b$  should be the same as the ratio  $\frac{1}{2} : \mu$

$$a = \frac{\frac{1}{2}}{\frac{1}{2} + \mu} h \quad b = \frac{\mu}{\frac{1}{2} + \mu} h$$

## MAXIMIZATION

If we know the expected values of  $a$  and  $b$  we could compute the maximum likelihood value of  $\mu$

$$\mu = \frac{b + c}{6(b + c + d)}$$

We begin with a guess for  $\mu$

We iterate between EXPECTATION and MAXIMALIZATION to improve our estimates of  $\mu$  and  $b$ .

Define  $\mu(t)$  the estimate of  $\mu$  on the  $t$ 'th iteration

$b(t)$  the estimate of  $b$  on  $t$ 'th iteration

REMEMBER

$$P(A) = \frac{1}{2}$$

$$P(B) = \mu$$

$$P(C) = 2\mu$$

$$P(D) = \frac{1}{2} - 3\mu$$

$\mu(0)$  = initial guess

$$b(t) = \frac{\mu(t)h}{\frac{1}{2} + \mu(t)} = E[b \mid \mu(t)]$$

$$\mu(t+1) = \frac{b(t) + c}{6(b(t) + c + d)}$$

= max like est of  $\mu$  given  $b(t)$



**Continue iterating until converged.**

**Good news: Converging to local optimum is assured.**

**Bad news: I said "local" optimum.**

- Convergence proof based on fact that  $\text{Prob}(\text{data} \mid \mu)$  must increase or remain same between each iteration [NOT OBVIOUS]
- But it can never exceed 1 [OBVIOUS]

So it must therefore converge [OBVIOUS]

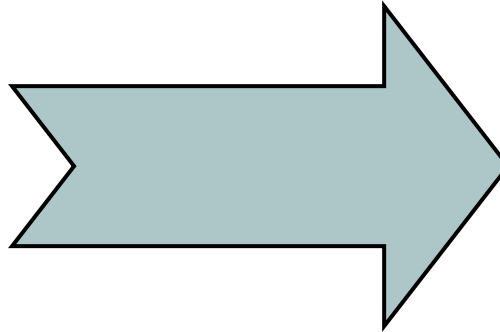
In our example, suppose we had

$h = 20$

$c = 10$

$d = 10$

$\mu(0) = 0$



Convergence is generally linear: error decreases by a constant factor each time step.

$t$	$\mu(t)$	$b(t)$
0	0	0
1	0.0833	2.857
2	0.0937	3.158
3	0.0947	3.185
4	0.0948	3.187
5	0.0948	3.187
6	0.0948	3.187

Remember:

We have unlabeled data  $x_1 x_2 \dots x_R$

We know there are  $k$  classes

We know  $P(w_1) P(w_2) P(w_3) \dots P(w_k)$

We don't know  $\mu_1 \mu_2 \dots \mu_k$

We can write  $P(\text{data} \mid \mu_1 \dots \mu_k)$

$$= p(x_1 \dots x_R \mid \mu_1 \dots \mu_k)$$

$$= \prod_{i=1}^R p(x_i \mid \mu_1 \dots \mu_k)$$

$$= \prod_{i=1}^R \sum_{j=1}^k p(x_i \mid w_j, \mu_1 \dots \mu_k) P(w_j)$$

$$= \prod_{i=1}^R \sum_{j=1}^k K \exp\left(-\frac{1}{2\sigma^2} (x_i - \mu_j)^2\right) P(w_j)$$

For Max likelihood we know  $\frac{\partial}{\partial \mu_i} \log \text{Prob}(\text{data} | \mu_1 \dots \mu_k) = 0$

Some wild'n'crazy algebra turns this into : "For Max likelihood, for each  $j$ ,

$$\mu_j = \frac{\sum_{i=1}^R P(w_j | x_i, \mu_1 \dots \mu_k) x_i}{\sum_{i=1}^R P(w_j | x_i, \mu_1 \dots \mu_k)}$$

This is  $n$  nonlinear equations in  $\mu_j$ 's."

If, for each  $x_i$  we knew that for each  $w_j$  the prob that  $\mu_j$  was in class  $w_j$  is  $P(w_j | x_i, \mu_1 \dots \mu_k)$  Then... we would easily compute  $\mu_j$ .

If we knew each  $\mu_j$  then we could easily compute  $P(w_j | x_i, \mu_1 \dots \mu_k)$  for each  $w_j$  and  $x_i$ .

...I feel an EM experience coming on!!

Iterate. On the  $t$ 'th iteration let our estimates be

$$\lambda_t = \{ \mu_1(t), \mu_2(t) \dots \mu_c(t) \}$$

## E-step

Compute “expected” classes of all datapoints for each class

$$P(w_i | x_k, \lambda_t) = \frac{p(x_k | w_i, \lambda_t) P(w_i | \lambda_t)}{p(x_k | \lambda_t)} = \frac{p(x_k | w_i, \mu_i(t), \sigma^2 \mathbf{I}) p_i(t)}{\sum_{j=1}^c p(x_k | w_j, \mu_j(t), \sigma^2 \mathbf{I}) p_j(t)}$$

*Just evaluate a Gaussian at  $x_k$*

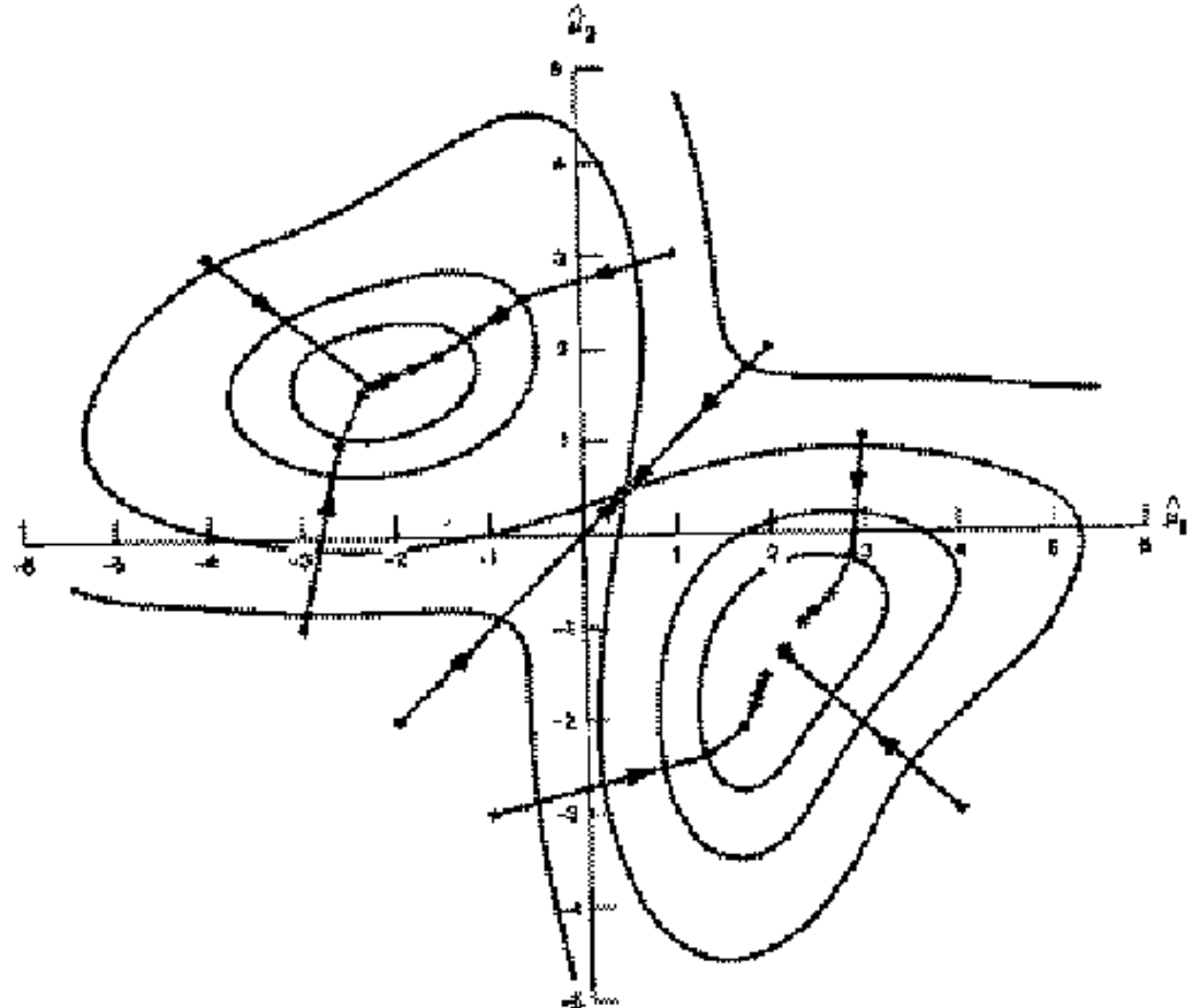
M-step.

Compute Max. like  $\mu$  given our data's class membership distributions

$$\mu_i(t+1) = \frac{\sum_k P(w_i | x_k, \lambda_t) x_k}{\sum_k P(w_i | x_k, \lambda_t)}$$



- This algorithm is **REALLY USED**. And in high dimensional state spaces, too. E.G. Vector Quantization for Speech Data
  - Your lecturer will (unless out of time) give you a nice intuitive explanation of why this rule works.
  - As with all EM procedures, convergence to a local optimum guaranteed.



Iterate. On the  $t$ 'th iteration let our estimates be

$$\lambda_t = \{ \mu_1(t), \mu_2(t) \dots \mu_c(t), \Sigma_1(t), \Sigma_2(t) \dots \Sigma_c(t), p_1(t), p_2(t) \dots p_c(t) \}$$

$p_i(t)$  is shorthand for estimate of  $P(\omega_i)$  on  $t$ 'th iteration

## E-step

Compute “expected” classes of all datapoints for each class

Just evaluate a Gaussian at  $x_k$

$$P(w_i | x_k, \lambda_t) = \frac{p(x_k | w_i, \lambda_t) P(w_i | \lambda_t)}{p(x_k | \lambda_t)} = \frac{p(x_k | w_i, \mu_i(t), \Sigma_i(t)) p_i(t)}{\sum_{j=1}^c p(x_k | w_j, \mu_j(t), \Sigma_j(t)) p_j(t)}$$

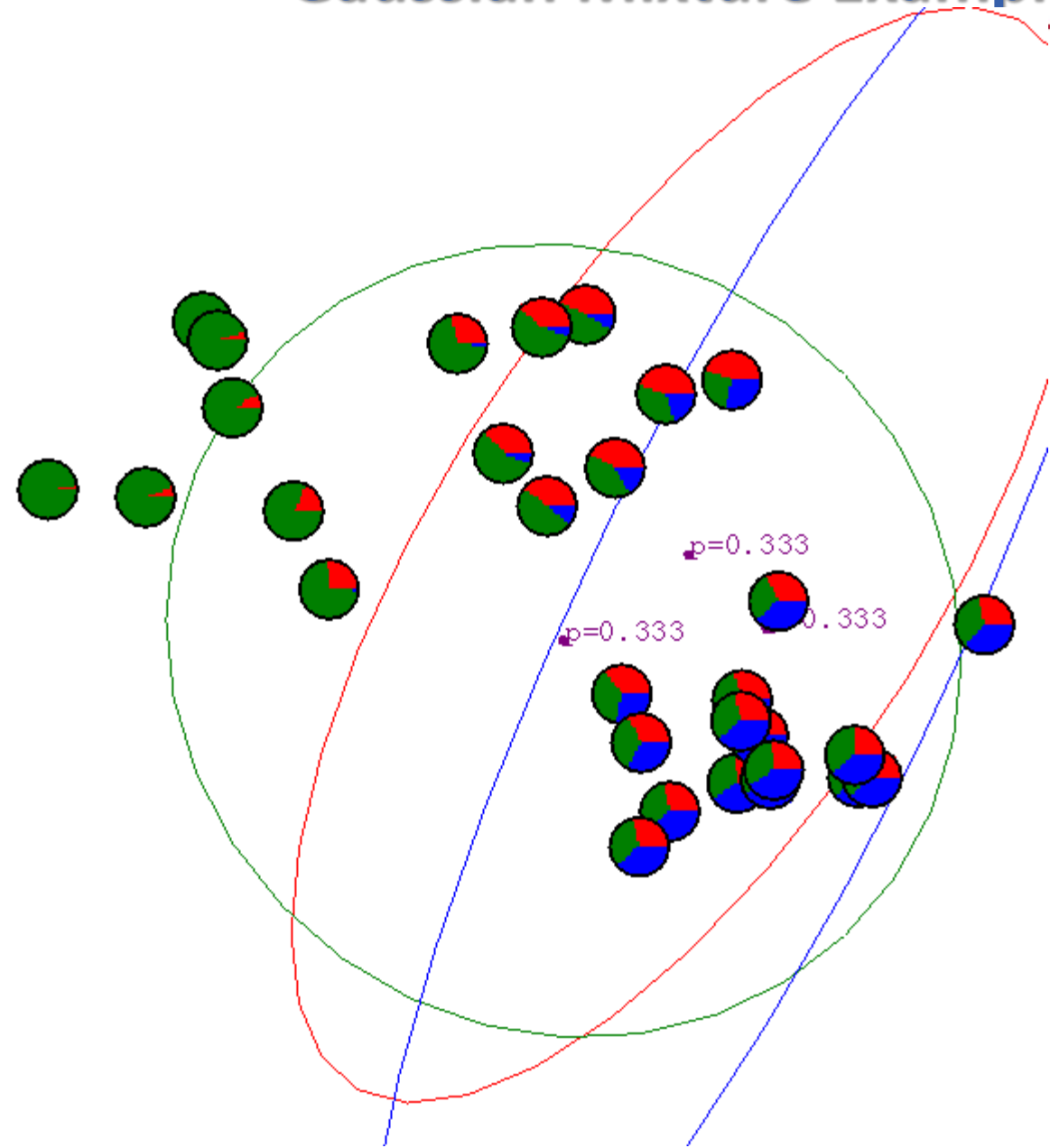
M-step.

Compute Max. like  $\mu$  given our data's class membership distributions

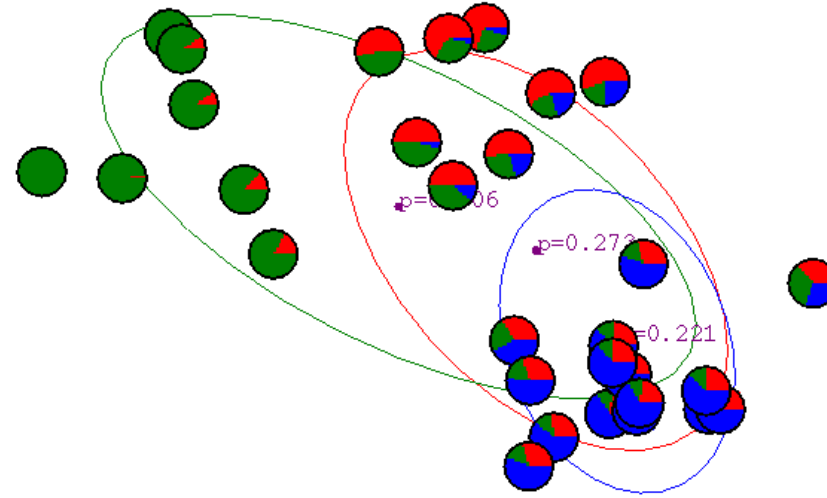
$$\mu_i(t+1) = \frac{\sum_k P(w_i | x_k, \lambda_t) x_k}{\sum_k P(w_i | x_k, \lambda_t)} \quad \Sigma_i(t+1) = \frac{\sum_k P(w_i | x_k, \lambda_t) [x_k - \mu_i(t+1)][x_k - \mu_i(t+1)]^T}{\sum_k P(w_i | x_k, \lambda_t)}$$

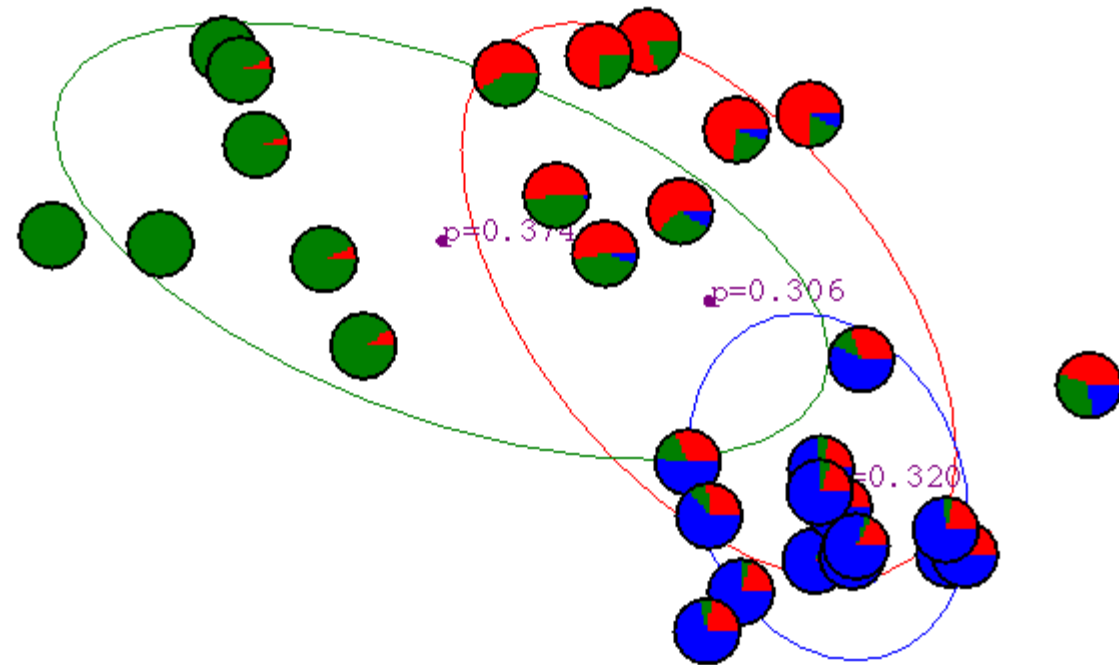
$$p_i(t+1) = \frac{\sum_k P(w_i | x_k, \lambda_t)}{R}$$

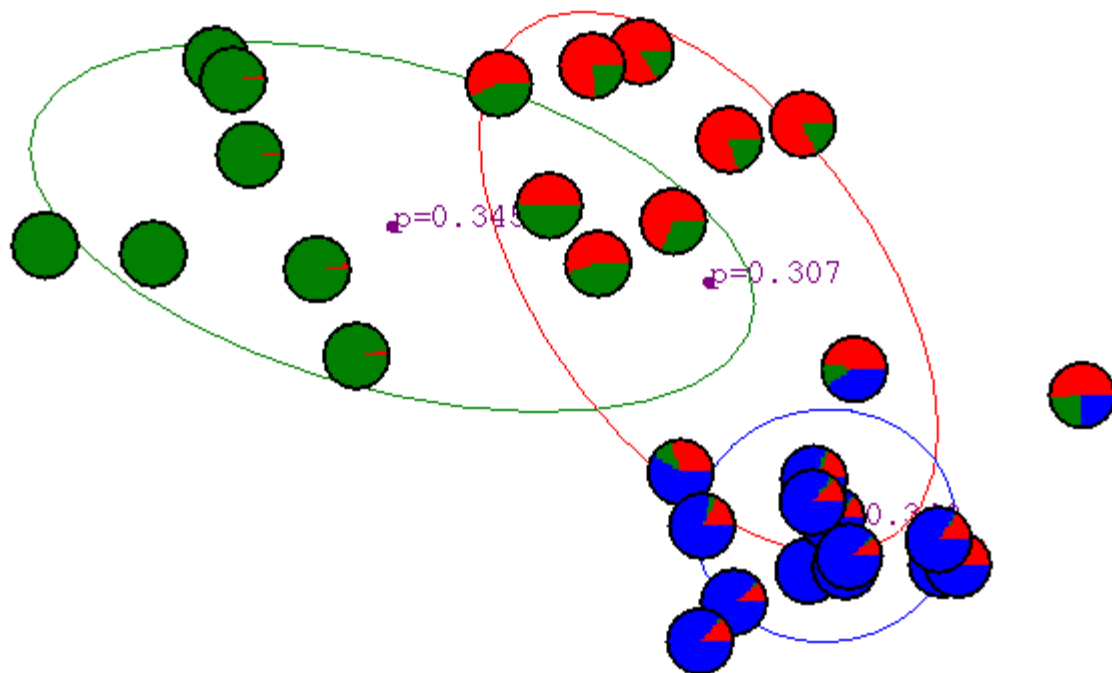
$R = \text{\#records}$

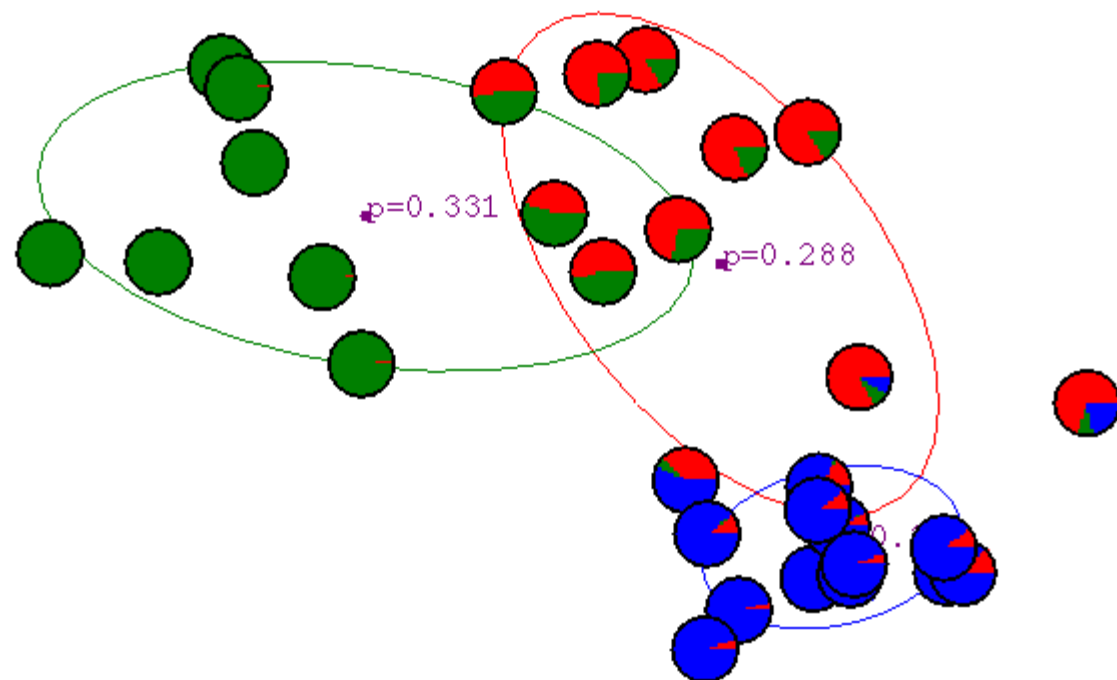


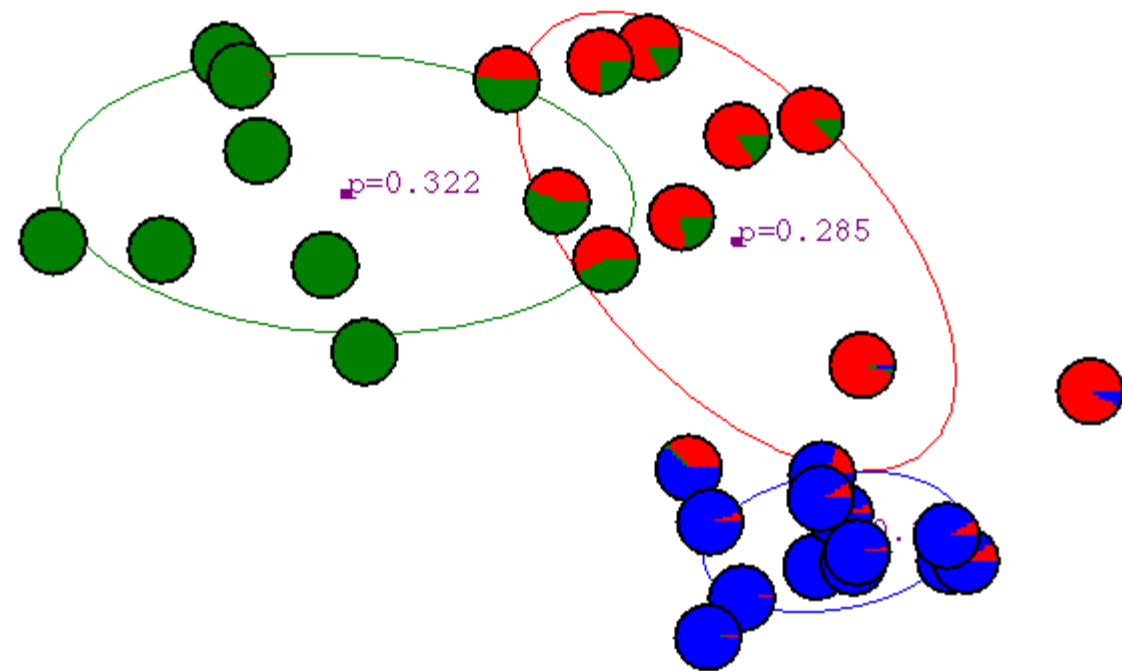
*Advance apologies: in Black and White this example will be incomprehensible*



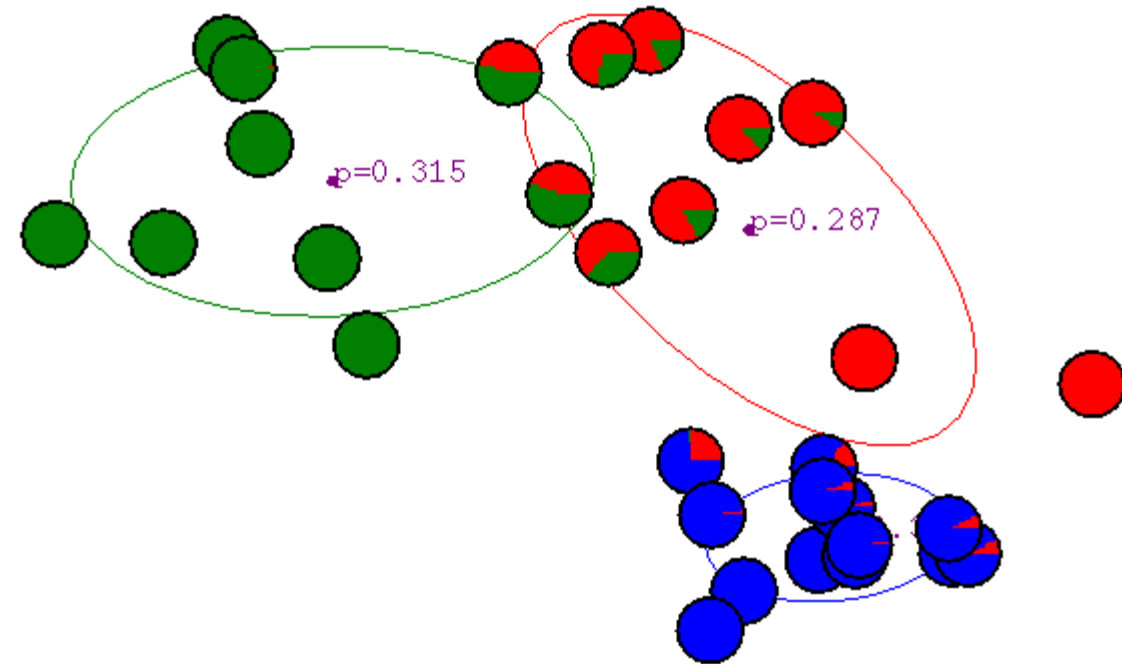


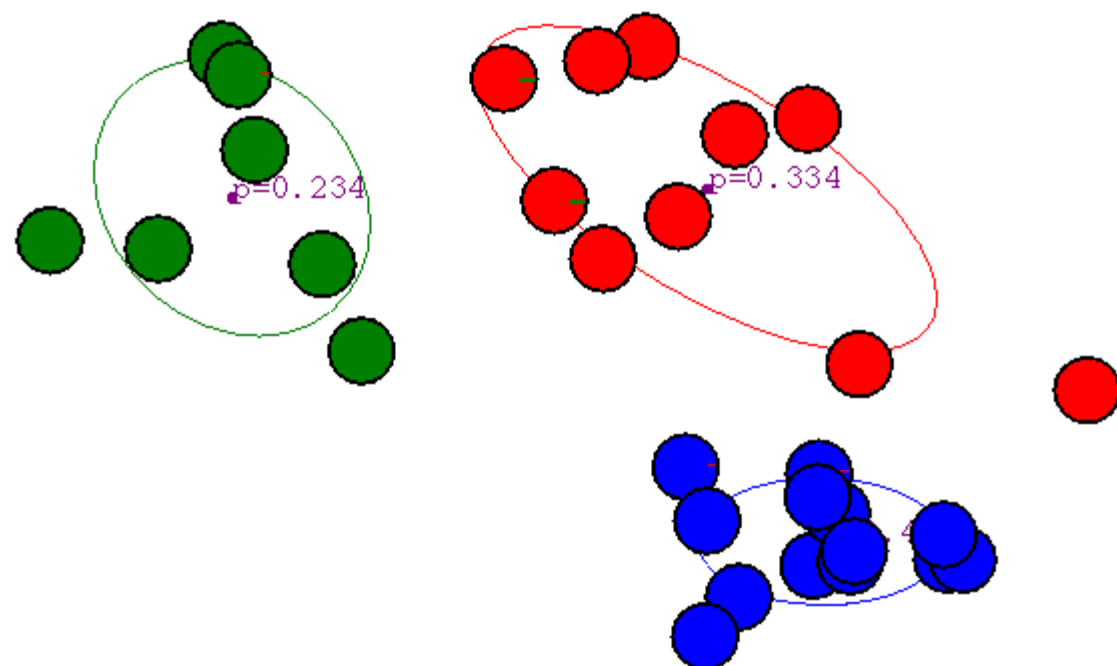


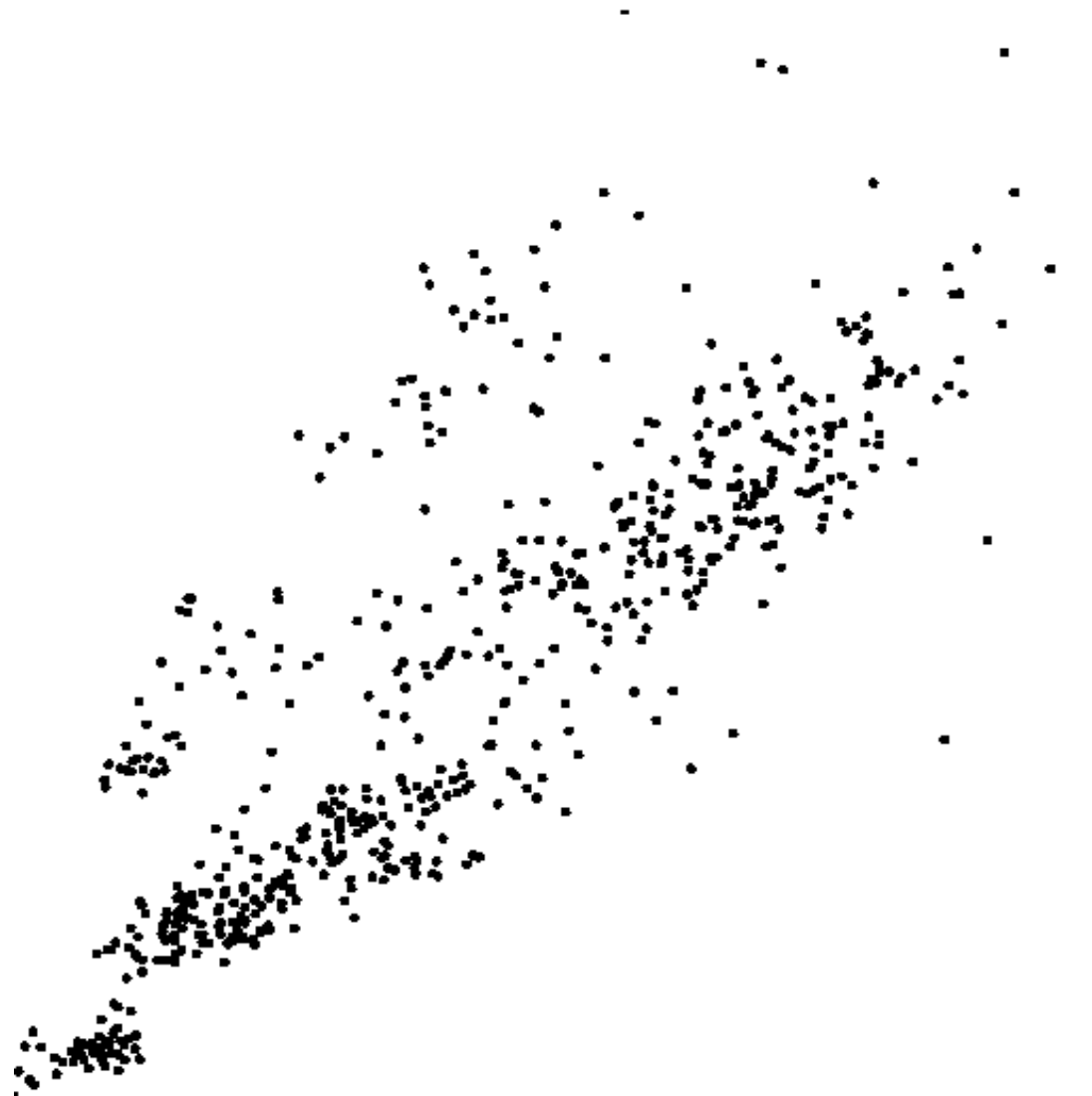


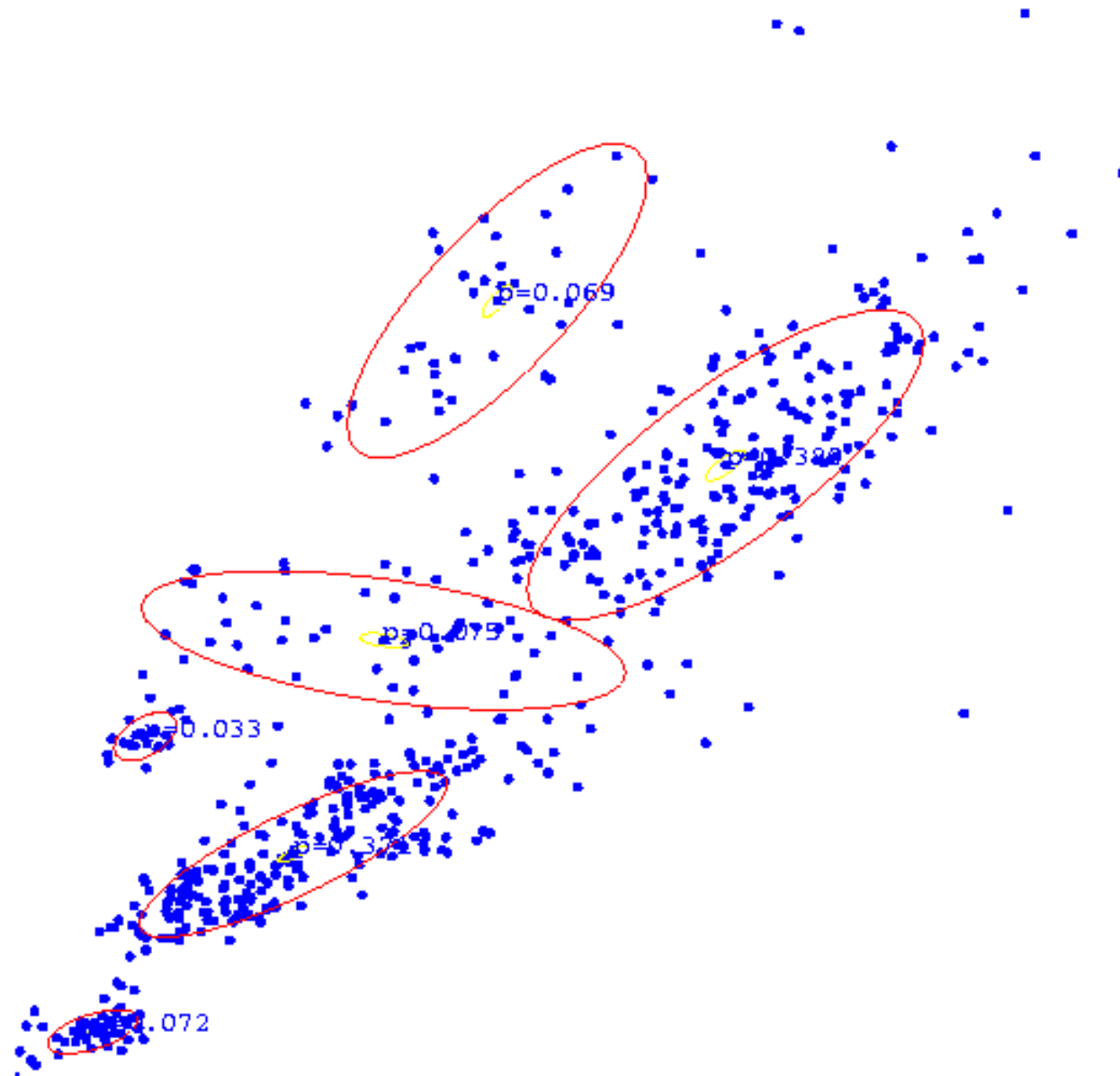


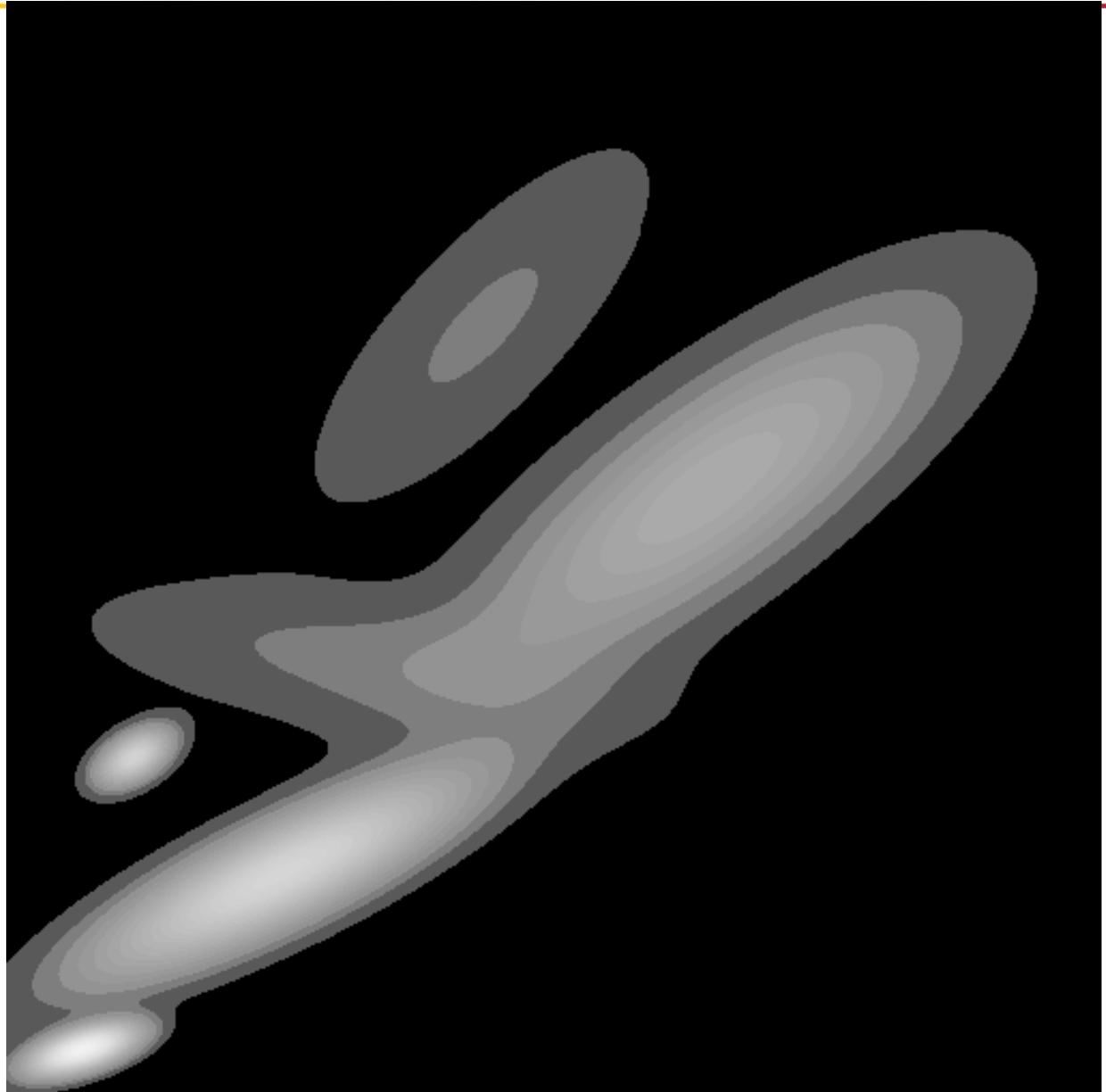






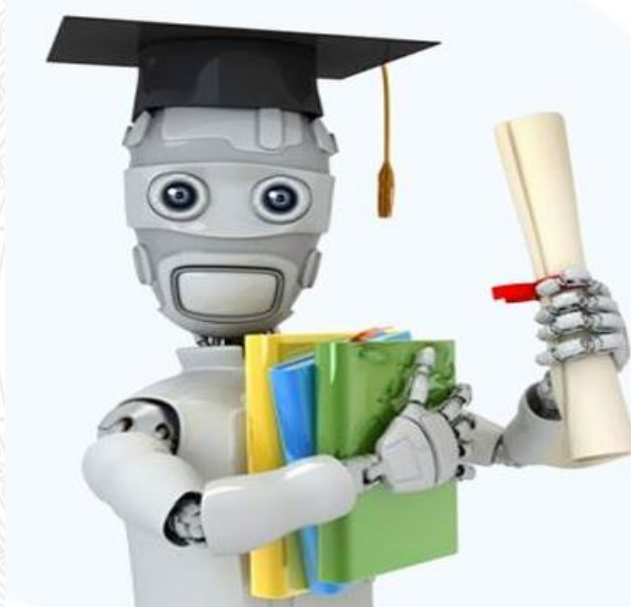






- Clustering Problems
- K-Means
- DBSCAN
- Gaussian Mixtures





**Enjoy the Course...!**