



# COL333/671: Introduction to AI

Semester I, 2024-25

## Learning with Probabilities

Rohan Paul

# Outline

- Last Class
  - CSPs
- This Class
  - Bayesian Learning, MLE/MAP, Learning in Probabilistic Models.
- Reference Material
  - Please follow the notes as the primary reference on this topic. Supplementary reading on topics covered in class from AIMA Ch 20 sections 20.1 – 20.2.4.

# Acknowledgement

**These slides are intended for teaching purposes only. Some material has been used/adapted from web sources and from slides by Doina Precup, Dorsa Sadigh, Percy Liang, Mausam, Parag, Emma Brunskill, Alexander Amini, Dan Klein, Anca Dragan, Nicholas Roy and others.**

# Learning Probabilistic Models

- Models are useful for making optimal decisions.
  - Probabilistic models express a theory about the domain and can be used for decision making.
- How to acquire these models in the first place?
  - Solution: data or experience can be used to build these models
- Key question: how to learn from data?
  - Bayesian view of learning (learning task itself is probabilistic inference)
  - Learning with complete and incomplete data.
  - Essentially, rely on counting.

# Example: *Which candy bag is it?*

Suppose there are five kinds of bags of candies:

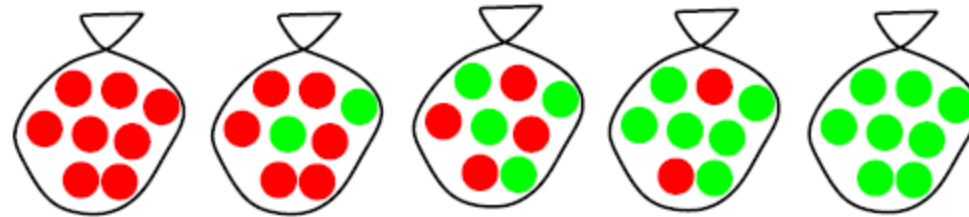
10% are  $h_1$ : 100% cherry candies

20% are  $h_2$ : 75% cherry candies + 25% lime candies

40% are  $h_3$ : 50% cherry candies + 50% lime candies

20% are  $h_4$ : 25% cherry candies + 75% lime candies

10% are  $h_5$ : 100% lime candies



Then we observe candies drawn from some bag: ● ● ● ● ● ● ● ● ● ●

What kind of bag is it? What flavour will the next candy be?

Statistics

Probability

# Bayesian Learning – in a nutshell

View learning as Bayesian updating of a probability distribution over the **hypothesis space**

$H$  is the hypothesis variable, values  $h_1, h_2, \dots$ , prior  $\mathbf{P}(H)$

$i$ th observation  $x_i$  gives the outcome of random variable  $X_i$

training data  $\mathbf{X} = x_1, \dots, x_N$

Given the data so far, each hypothesis has a posterior probability:

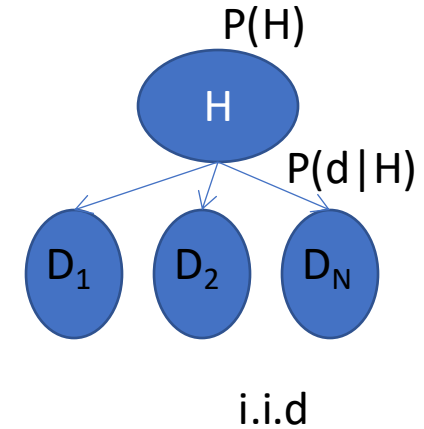
$$P(h_k|\mathbf{X}) = \alpha P(\mathbf{X}|h_k)P(h_k)$$

where  $P(\mathbf{X}|h_k)$  is called the **likelihood**

Predictions use a likelihood-weighted average over the hypotheses:

$$\mathbf{P}(X_{N+1}|\mathbf{X}) = \sum_k \mathbf{P}(X_{N+1}|\mathbf{X}, h_k)P(h_k|\mathbf{X}) = \sum_k \mathbf{P}(X_{N+1}|h_k)P(h_k|\mathbf{X})$$

No need to pick one best-guess hypothesis!



In these slides  $X$  and  $d$  used interchangeably.

# Posterior Probability of Hypothesis given Observations

Now, we are getting observations incrementally, how does our belief change?

Suppose there are five kinds of bags of candies:

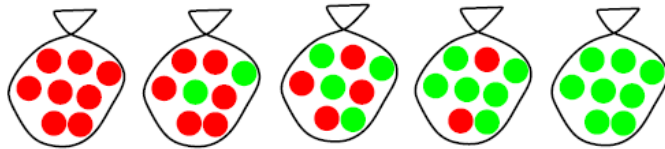
10% are  $h_1$ : 100% cherry candies

20% are  $h_2$ : 75% cherry candies + 25% lime candies

40% are  $h_3$ : 50% cherry candies + 50% lime candies

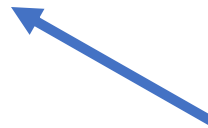
20% are  $h_4$ : 25% cherry candies + 75% lime candies

10% are  $h_5$ : 100% lime candies



Then we observe candies drawn from some bag: ●●●●●●●●●●

What kind of bag is it? What flavour will the next candy be?



Probability of a bag of a certain type given observations.

$$P(h_i | d_1, \dots, d_N)$$

Bayes Rule

$$P(h_i | \mathbf{d}) = \alpha P(\mathbf{d} | h_i) P(h_i)$$

IID assumption

$$P(\mathbf{d} | h_i) = \prod_j P(d_j | h_i)$$

# Posterior Probability of Hypothesis given Observations

$$P(h_k|\mathbf{X}) = \alpha P(\mathbf{X}|h_k)P(h_k)$$

$$P(h_1 | 5 \text{ limes}) = \alpha P(5 \text{ limes} | h_1)P(h_1) = \alpha \cdot 0.0^5 \cdot 0.1 = 0$$

$$P(h_2 | 5 \text{ limes}) = \alpha P(5 \text{ limes} | h_2)P(h_2) = \alpha \cdot 0.25^5 \cdot 0.2 = 0.000195\alpha$$

$$P(h_3 | 5 \text{ limes}) = \alpha P(5 \text{ limes} | h_3)P(h_3) = \alpha \cdot 0.5^5 \cdot 0.4 = 0.0125\alpha$$

$$P(h_4 | 5 \text{ limes}) = \alpha P(5 \text{ limes} | h_4)P(h_4) = \alpha \cdot 0.75^5 \cdot 0.2 = 0.0475\alpha$$

$$P(h_5 | 5 \text{ limes}) = \alpha P(5 \text{ limes} | h_5)P(h_5) = \alpha \cdot 1.0^5 \cdot 0.1 = 0.1\alpha$$

$$\alpha = 1/(0 + 0.000195 + 0.0125 + 0.0475 + 0.1) = 6.2424$$

$$P(h_1 | 5 \text{ limes}) = 0$$

$$P(h_2 | 5 \text{ limes}) = 0.00122$$

$$P(h_3 | 5 \text{ limes}) = 0.07803$$

$$P(h_4 | 5 \text{ limes}) = 0.29650$$

$$P(h_5 | 5 \text{ limes}) = 0.62424$$

# Incremental Belief Update

Suppose there are five kinds of bags of candies:

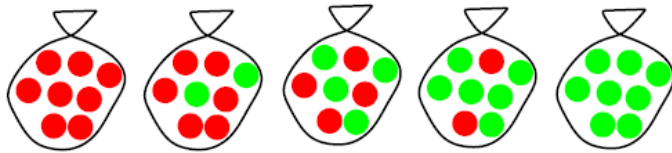
10% are  $h_1$ : 100% cherry candies

20% are  $h_2$ : 75% cherry candies + 25% lime candies

40% are  $h_3$ : 50% cherry candies + 50% lime candies

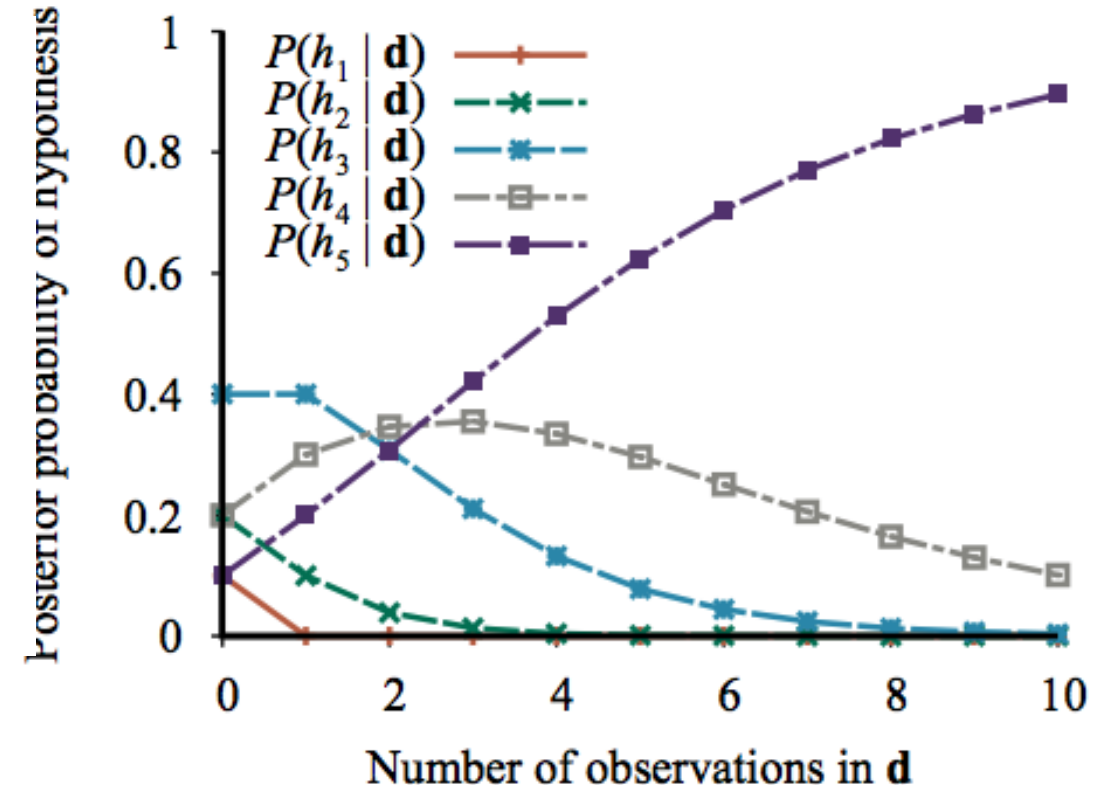
20% are  $h_4$ : 25% cherry candies + 75% lime candies

10% are  $h_5$ : 100% lime candies



Then we observe candies drawn from some bag: ● ● ● ● ● ● ● ● ● ●

What kind of bag is it? What flavour will the next candy be?



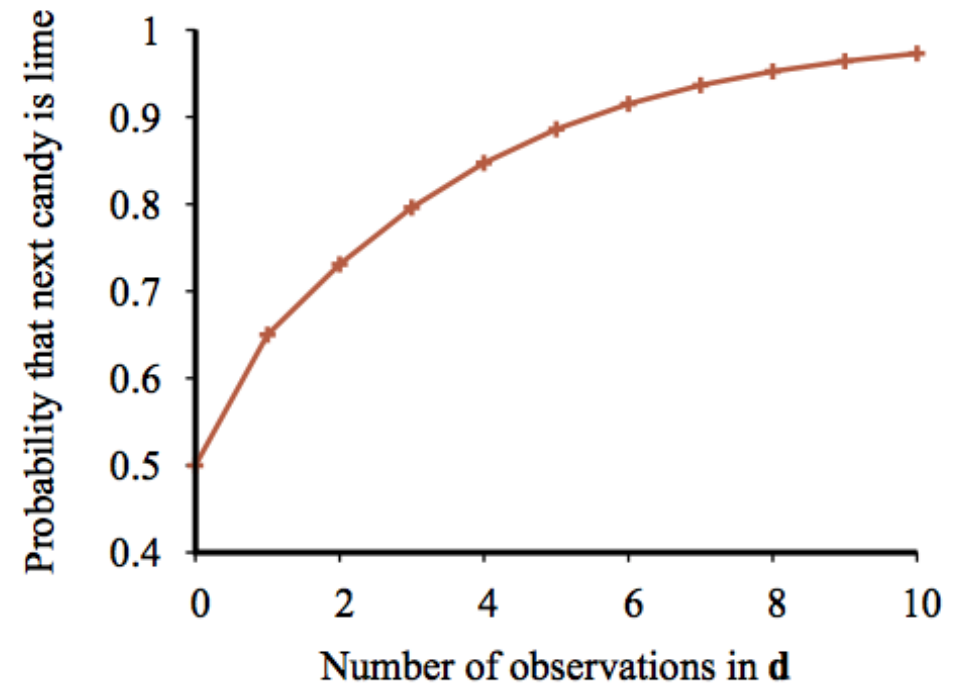
True hypothesis eventually dominates. Probability of indefinitely producing uncharacteristic data  $\rightarrow 0$

# Predictions given Belief over Hypotheses

What is the probability that the next candy is of type lime?

$$\mathbf{P}(X_{N+1}|\mathbf{X}) = \sum_k \mathbf{P}(X_{N+1}|\mathbf{X}, h_k)P(h_k|\mathbf{X}) = \sum_k \mathbf{P}(X_{N+1}|h_k)P(h_k|\mathbf{X})$$

$$\begin{aligned} P(\text{lime on 6} \mid 5 \text{ limes}) &= P(\text{lime on 6} \mid h_1)P(h_1 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_2)P(h_2 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_3)P(h_3 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_4)P(h_4 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_5)P(h_5 \mid 5 \text{ limes}) \\ &= 0 \times 0 \\ &+ 0.25 \times 0.00122 \\ &+ 0.5 \times 0.07830 \\ &+ 0.75 \times 0.29650 \\ &+ 1.0 \times 0.62424 \\ &= 0.88607 \end{aligned}$$



Observations



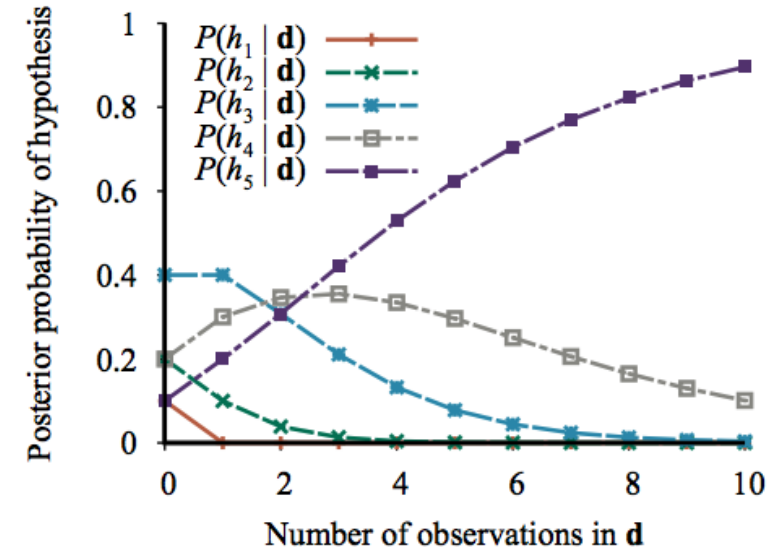
# Bayesian Prediction – key ideas

- Predictions are weighted average over the predictions of the individual hypothesis.
- Bayesian prediction eventually agrees with the true hypothesis.
- For any fixed prior that does not rule out the true hypothesis, the posterior probability of any false hypothesis will eventually vanish.
- Why keep all the hypothesis?
  - Learning from small data, early commitment to a hypothesis is risky, later evidence may lead to a different likely hypothesis.
  - Better accounting of uncertainty in making predictions.
  - Problem: maybe slow and intractable, cannot estimate and marginalize out the hypotheses.

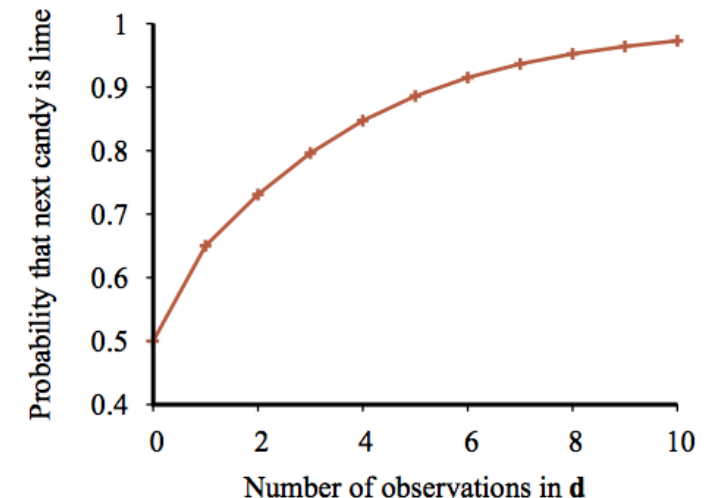
Evidence arrives incrementally



Changing belief



Prediction by model averaging.



# Marginalization over Hypothesis – *challenging!*

Ideally, one needs to marginalize or account for all the hypotheses.

Can we pick one good hypothesis and just use that for predications?

$$\mathbf{P}(X_{N+1}|\mathbf{X}) = \sum_k \mathbf{P}(X_{N+1}|\mathbf{X}, h_k)P(h_k|\mathbf{X}) = \sum_k \mathbf{P}(X_{N+1}|h_k)P(h_k|\mathbf{X})$$

$$\begin{aligned} P(\text{lime on 6} \mid 5 \text{ limes}) &= P(\text{lime on 6} \mid h_1)P(h_1 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_2)P(h_2 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_3)P(h_3 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_4)P(h_4 \mid 5 \text{ limes}) \\ &+ P(\text{lime on 6} \mid h_5)P(h_5 \mid 5 \text{ limes}) \end{aligned}$$

# Maximum a-posteriori (MAP) Approximation

Make predictions based on a **single most probable hypothesis**

$$P(X | d) \approx P(X | h_{MAP})$$

$$P(h_{MAP}) = \operatorname{argmax}_{h_i} (P(h_i | d))$$

$$P(h_{MAP}) = \operatorname{argmax}_{h_i} (P(d | h_i) P(h_i))$$

$$= \operatorname{argmax}_{h_i} (\log(P(d | h_i)) + \log(P(h_i)))$$

*What is the probability of a hypothesis given data?*

- MAP learning chooses the hypothesis that provides maximum *compression* of the data.
  - $\log_2 P(h_i)$ : the number of bits required to specify the hypothesis  $h_i$ .
  - $\log_2 P(d | h_i)$ : the additional number of bits required to specify the data, given the hypothesis.

Estimate the best hypothesis given data while incorporating the prior knowledge.

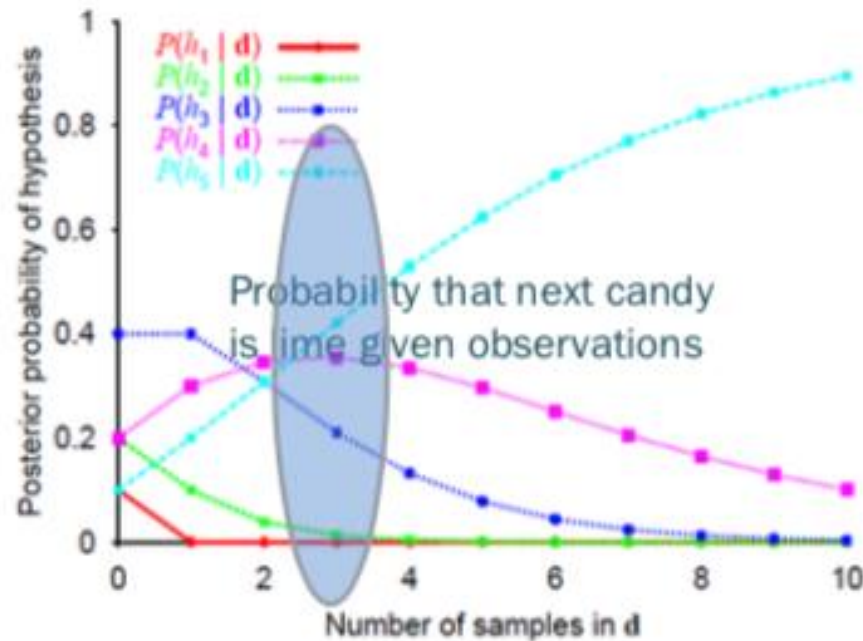
The prior term says which hypothesis are likelier than others. Typically, the number of bits to encode hypothesis.

# MAP Vs. Bayesian Estimation

EX> ● ● ● After three observations

MAP predict with probability 1 that next candy is lime  
(pick  $h_5$ )

Bayes will predict with probability 0.8 that net is lime



Difference between marginalization  
(accounting for all hypothesis) vs.  
committing to one and make  
predictions from it.

# Maximum Likelihood Estimation

Assume **uniform prior** over the space of hypothesis

MAP with uniform prior: Maximum-likelihood hypothesis

$$P(h_{MAP}) = \operatorname{argmax}_{h_i} (\log(P(d | h_i)) + \log(P(h_i)))$$

Becomes irrelevant if  
uniform

$$P(h_{ML}) = \operatorname{argmax}_{h_i} (\log(P(d | h_i)))$$

Make predictions with the hypothesis that maximizes the data likelihood. Essentially, assuming a uniform prior with no preference of a hypothesis over another.

MLE is also called Maximum likelihood (ML) Approximation

# Maximum Likelihood Approximation

For large data sets, prior becomes irrelevant

Maximum likelihood (ML) learning: choose  $h_{\text{ML}}$  maximizing  $P(\mathbf{X}|h_k)$

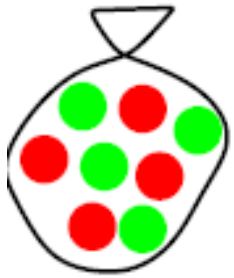
I.e., simply get the best fit to the data; identical to MAP for uniform prior (which is reasonable if all hypotheses are of the same complexity)

ML is the “standard” (non-Bayesian) statistical learning method

$$\begin{aligned}\theta_{ML} &= \arg \max_{\theta} P(\mathbf{X}|\theta) \\ &= \arg \max_{\theta} \prod_i P_{\theta}(X_i)\end{aligned}$$

# ML Estimation in General: Bernoulli Model

Hypothesis is the likelihood of generating a candy of a specific flavor.



Cherry, Lime, Lime, Cherry, Cherry,  
Lime, Cherry, Cherry

E.g., Bernoulli $[\theta]$  model:

$$P(X_i = 1) = \theta; \quad P(X_i = 0) = 1 - \theta$$

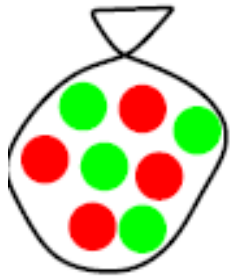
$$\text{or } P(X_i = x_i) = \theta^{x_i}(1 - \theta)^{1-x_i}$$

Suppose we get a bag of candy from a new manufacturer;  
fraction  $\theta$  of cherry candies

Any  $\theta$  is possible: continuum of hypotheses  $h_\theta$

Similar problem to observing tosses of a biased coin and estimating the bias/fractional parameter.

# ML Estimation in General: Estimation for Bernoulli Model



Cherry, Lime, Lime, Cherry, Cherry,  
Lime, Cherry, Cherry

Suppose we unwrap  $N$  candies,  $c$  cherries and  $\ell = N - c$  limes  
These are **i.i.d.** (independent, identically distributed) observations, so

$$P(\mathbf{X}|h_\theta) = \prod_{i=1}^N P(x_i|h_\theta) = \theta^{\sum_i x_i} (1 - \theta)^{N - \sum_i x_i} = \theta^c \cdot (1 - \theta)^\ell$$

Maximize this w.r.t.  $\theta$ —which is easier for the **log-likelihood**:

$$\begin{aligned} L(\mathbf{X}|h_\theta) &= \log P(\mathbf{X}|h_\theta) = \sum_{i=1}^N \log P(x_i|h_\theta) = c \log \theta + \ell \log(1 - \theta) \\ \frac{dL(\mathbf{X}|h_\theta)}{d\theta} &= \frac{c}{\theta} - \frac{\ell}{1 - \theta} = 0 \quad \Rightarrow \quad \theta = \frac{c}{c + \ell} = \frac{c}{N} \end{aligned}$$

Even in the coin tossing problem, one would take the fraction as heads or tails over the total number of tosses.

# MAP vs. MLE Estimation

- **Maximum likelihood estimate (MLE)**

- Estimates the parameters that maximizes the data likelihood.
- Relative counts give MLE estimates

$$\begin{aligned}\theta_{ML} &= \arg \max_{\theta} P(\mathbf{X}|\theta) \\ &= \arg \max_{\theta} \prod_i P_{\theta}(X_i)\end{aligned}$$

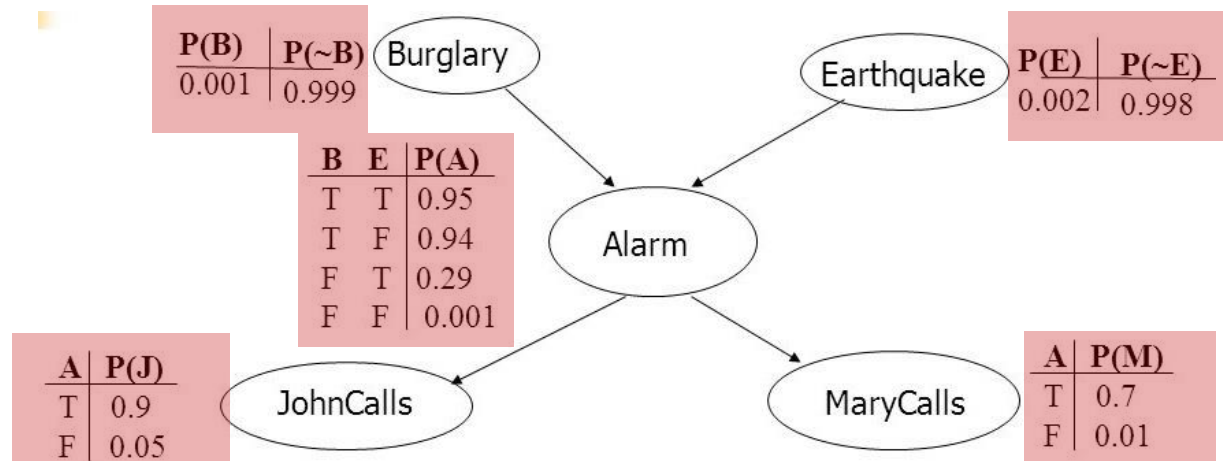
- **Maximum a posteriori estimate (MAP)**

- Bayesian parameter estimation
- Encodes a prior over the parameters (not all parameters are equal prior values).
- Combines the prior and the likelihood while estimating the parameters.

$$\begin{aligned}\theta_{MAP} &= \arg \max_{\theta} P(\theta|\mathbf{X}) \\ &= \arg \max_{\theta} P(\mathbf{X}|\theta)P(\theta)/P(\mathbf{X}) \\ &= \arg \max_{\theta} P(\mathbf{X}|\theta)P(\theta)\end{aligned}$$

# ML Estimation in General: Learning Parameters for a Probability Model

- Probabilistic models require parameters (numbers in the conditional probability tables).
- We need these values to make predictions.
- Can we learn these from data (i.e., samples from the Bayes Net)?
- How to do this? Counting and averaging.



Can we use samples to estimate the values in the tables?

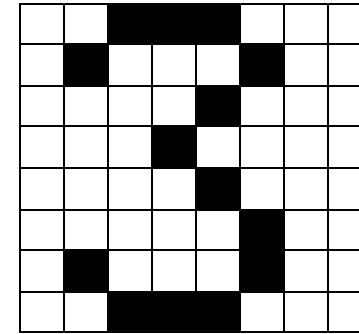
# Learning Parameters for a Probability Model

## Classification Problem

- Task: given inputs  $x$ , predict labels (classes)  $y$
- Examples:
  - Spam detection (input: document, classes: spam / ham)
  - OCR (input: images, classes: characters)
  - Medical diagnosis (input: symptoms, classes: diseases)
  - Fraud detection (input: account activity, classes: fraud / no fraud)

# Bayes Net for Classification

- Input: images / pixel grids
- Output: a digit 0-9
- Setup:
  - Get a large collection of example images, each labeled with a digit
  - Note: someone has to hand label all this data!
  - Want to learn to predict labels of new, future digit images
- Features: The attributes used to make the digit decision
  - Pixels: (6,8)=ON
  - Shape Patterns: NumComponents, AspectRatio, NumLoops
  - ...



0



1



2



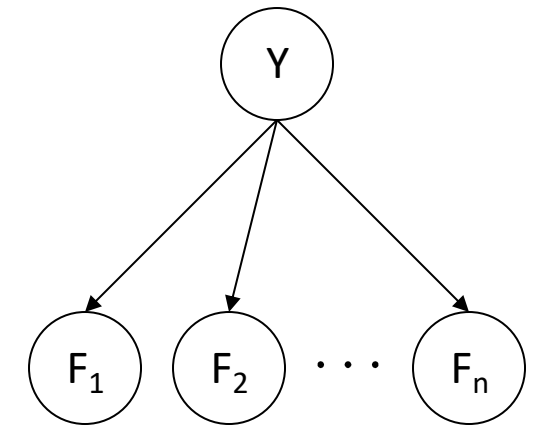
1



Not clear

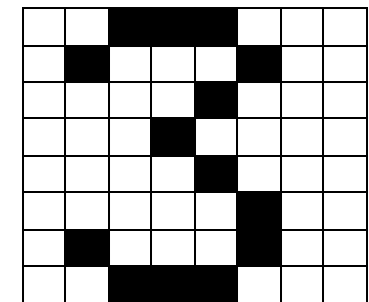
# Bayes Net for Classification

- Naïve Bayes: Assume all features are independent effects of the label
- Simple digit recognition:
  - One feature (variable)  $F_{ij}$  for each grid position  $\langle i, j \rangle$
  - Feature values are on / off, based on whether intensity is more or less than 0.5 in underlying image
  - Each input maps to a feature vector, e.g.



→  $\langle F_{0,0} = 0 \ F_{0,1} = 0 \ F_{0,2} = 1 \ F_{0,3} = 1 \ F_{0,4} = 0 \ \dots F_{15,15} = 0 \rangle$

$$P(Y|F_{0,0} \dots F_{15,15}) \propto P(Y) \prod_{i,j} P(F_{i,j}|Y)$$

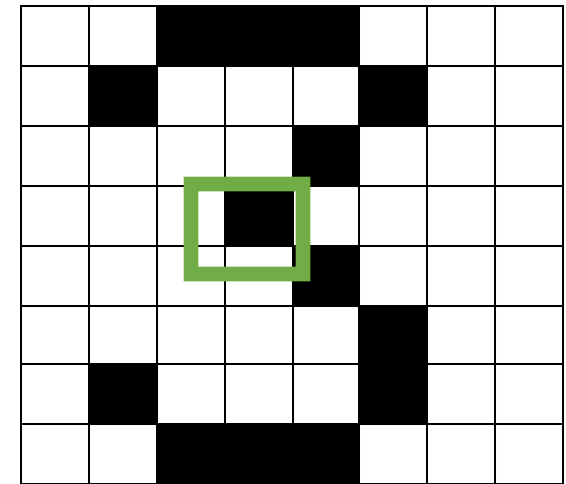


# Parameter Estimation

- Need the estimates of local conditional probability tables.
  - $P(Y)$ , the prior over labels
  - $P(F_i | Y)$  for each feature (evidence variable)
  - These probabilities are collectively called the *parameters* of the model and denoted by  $\theta$ 
    - Till now, the table values were provided.
    - Now, use data to acquire these values.

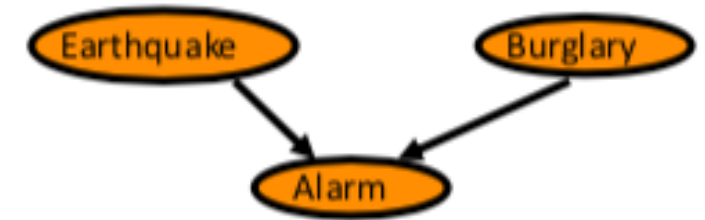
# Parameter Estimation

- $P(Y)$  – how frequent is the class-type for digit 3?
  - If you take a sample of images of numbers how frequent is this number
- $P(F_i|Y)$  – for digit 3 what fraction of the time the cell is on?
  - Conditioned on the class type how frequent is the feature
- Use **relative frequencies** from the data to estimate these values.



# Parameter Estimation: Complete Data

E	B	A	#
0	0	0	1000
0	0	1	10
0	1	0	20
0	1	1	100
1	0	0	200
1	0	1	50
1	1	0	0
1	1	1	5



Note: The data is “complete”. Each data point had values observed for “all” the variables in the model.

	Pr(A E,B)
e,b	
e, $\bar{b}$	
$\bar{e}$ ,b	
$\bar{e}$ , $\bar{b}$	

$$P(a|\bar{e},\bar{b}) = ?$$

$$= 10/1010$$

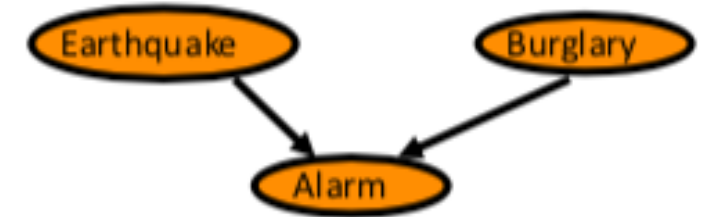
# Parameter Estimation

E	B	A	#
0	0	0	1000
0	0	1	10
0	1	0	20
0	1	1	100
1	0	0	200
1	0	1	50
1	1	0	0
1	1	1	5

	Pr(A E,B)
e,b	
e, $\bar{b}$	
$\bar{e}$ ,b	
$\bar{e}$ , $\bar{b}$	~0.01

$$P(a|\bar{e}, b) = ?$$

$$= 100/120$$



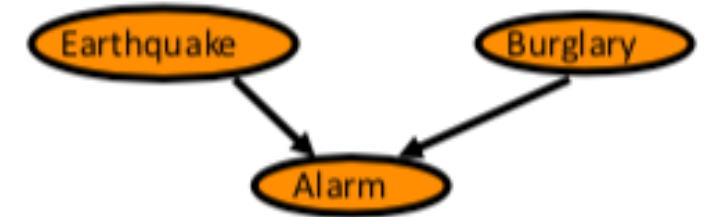
# Parameter Estimation

E	B	A	#
0	0	0	1000
0	0	1	10
0	1	0	20
0	1	1	100
1	0	0	200
1	0	1	50
1	1	0	0
1	1	1	5

	Pr(A E,B)
e,b	
e, $\bar{b}$	
$\bar{e}$ ,b	0.83
$\bar{e}$ , $\bar{b}$	~0.01

$$P(a|e, \bar{b}) = ?$$

$$= 50/250$$



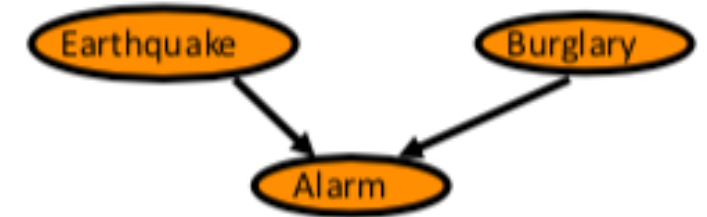
# Parameter Estimation

E	B	A	#
0	0	0	1000
0	0	1	10
0	1	0	20
0	1	1	100
1	0	0	200
1	0	1	50
1	1	0	0
1	1	1	5

	Pr(A E,B)
e,b	
e, $\bar{b}$	0.2
$\bar{e}$ ,b	0.83
$\bar{e}$ , $\bar{b}$	~0.01

$$P(a|e, b) = ?$$

$$= 5/5$$



# Problem: values not seen in the training data

$$P(\text{features}, C = 2)$$

$$P(C = 2) = 0.1$$

$$P(\text{on}|C = 2) = 0.8$$

$$P(\text{on}|C = 2) = 0.1$$

$$P(\text{off}|C = 2) = 0.1$$

$$P(\text{on}|C = 2) = 0.01$$

$$P(\text{features}, C = 3)$$

$$P(C = 3) = 0.1$$

$$P(\text{on}|C = 3) = 0.8$$

$$P(\text{on}|C = 3) = 0.9$$

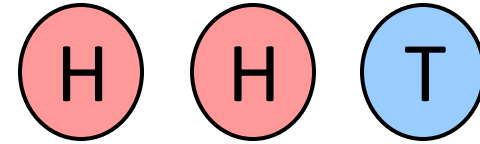
$$P(\text{off}|C = 3) = 0.7$$

$$P(\text{on}|C = 3) = 0.0$$

If one feature was not seen in the training data, the likelihood goes to zero. If we did not see this feature in the training data, does not mean we will not see this in training. Essentially overfitting to the training data set.

# Laplace Smoothing

- Pretend that every outcome occurs **once more** than it is observed.
- If certain counts are **not** seen in training does not mean that they have zero probability of occurring in future.
- Another version of Laplace smoothing
  - instead of 1, add **k times**
  - k is an adjustable parameter.
- Essentially, encodes a **prior** (pseudo-counts).



$$\begin{aligned} P_{LAP}(x) &= \frac{c(x) + 1}{\sum_x [c(x) + 1]} \\ &= \frac{c(x) + 1}{N + |X|} \end{aligned}$$

$$P_{LAP,k}(x) = \frac{c(x) + k}{N + k|X|}$$

# Learning Multiple Parameters

- Estimate latent parameters using MLE.
- There are two CPTs in this example.
- Observations are of both variables: Flavor and Wrapper.
- Take log likelihood.

Red/green wrapper depends probabilistically on flavor:

Likelihood for, e.g., cherry candy in green wrapper:

$$\begin{aligned} P(F = \text{cherry}, W = \text{green} | h_{\theta, \theta_1, \theta_2}) \\ &= P(F = \text{cherry} | h_{\theta, \theta_1, \theta_2}) P(W = \text{green} | F = \text{cherry}, h_{\theta, \theta_1, \theta_2}) \\ &= \theta \cdot (1 - \theta_1) \end{aligned}$$

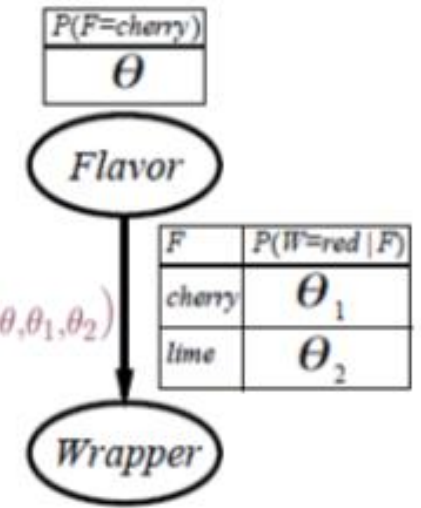
$N$  candies,  $r_c$  red-wrapped cherry candies, etc.:

$$P(d | h_{\theta, \theta_1, \theta_2}) = \theta^c (1 - \theta)^\ell \cdot \theta_1^{r_c} (1 - \theta_1)^{g_c} \cdot \theta_2^{r_\ell} (1 - \theta_2)^{g_\ell}$$

Take logarithm

$$\begin{aligned} L = & [c \log \theta + \ell \log(1 - \theta)] \\ & + [r_c \log \theta_1 + g_c \log(1 - \theta_1)] \\ & + [r_\ell \log \theta_2 + g_\ell \log(1 - \theta_2)] \end{aligned}$$

With complete data, the ML parameter learning problem for a Bayesian network decomposes into separate learning problems, one for each parameter



$N$  candies unwrapped,  $c$  are cherries and  $\ell$  are limes

# Learning Multiple Parameters

- Minimize data likelihood to estimate the parameters.

Derivatives of  $L$  contain only the relevant parameter:

$$\frac{\partial L}{\partial \theta} = \frac{c}{\theta} - \frac{\ell}{1 - \theta} = 0 \quad \Rightarrow \quad \theta = \frac{c}{c + \ell}$$

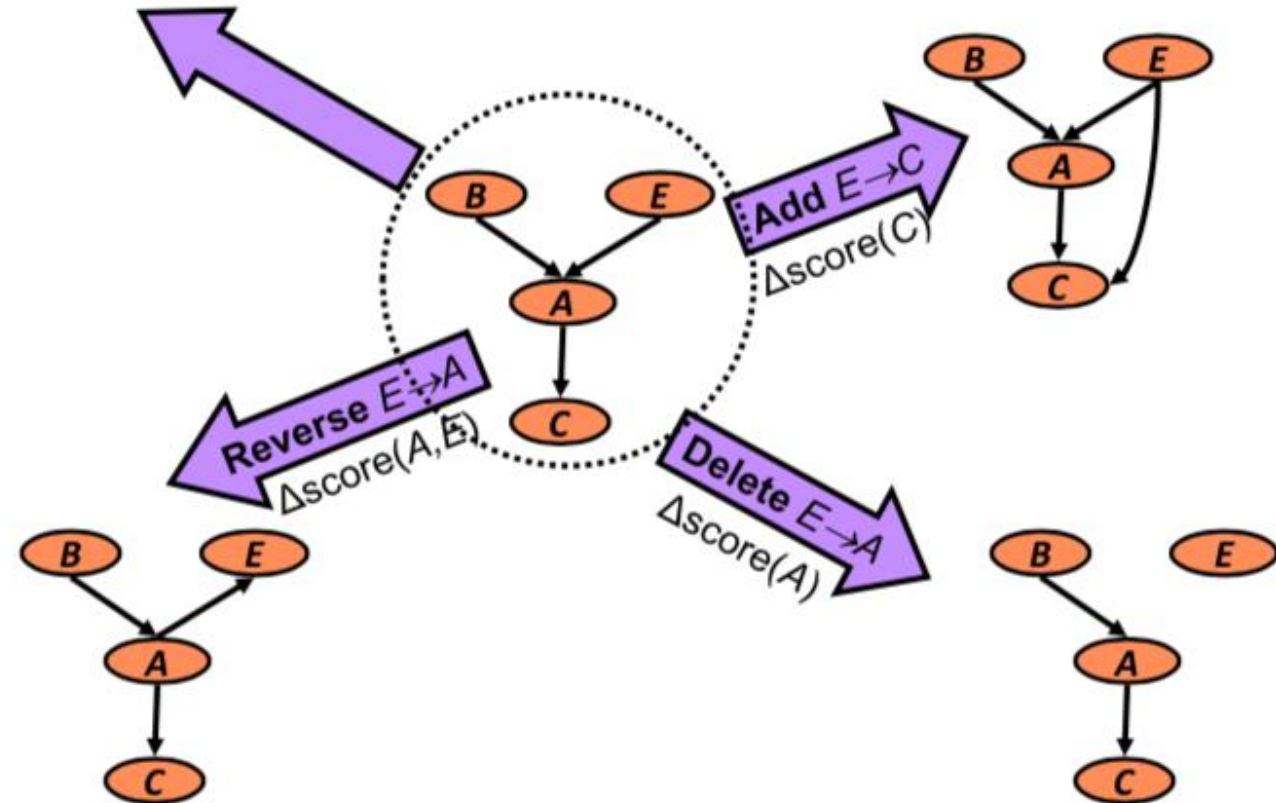
$$\frac{\partial L}{\partial \theta_1} = \frac{r_c}{\theta_1} - \frac{g_c}{1 - \theta_1} = 0 \quad \Rightarrow \quad \theta_1 = \frac{r_c}{r_c + g_c}$$

$$\frac{\partial L}{\partial \theta_2} = \frac{r_\ell}{\theta_2} - \frac{g_\ell}{1 - \theta_2} = 0 \quad \Rightarrow \quad \theta_2 = \frac{r_\ell}{r_\ell + g_\ell}$$

Maximum Likelihood Parameter Learning with complete data for a Bayes Net decomposes into separate learning problems, one for each parameter.

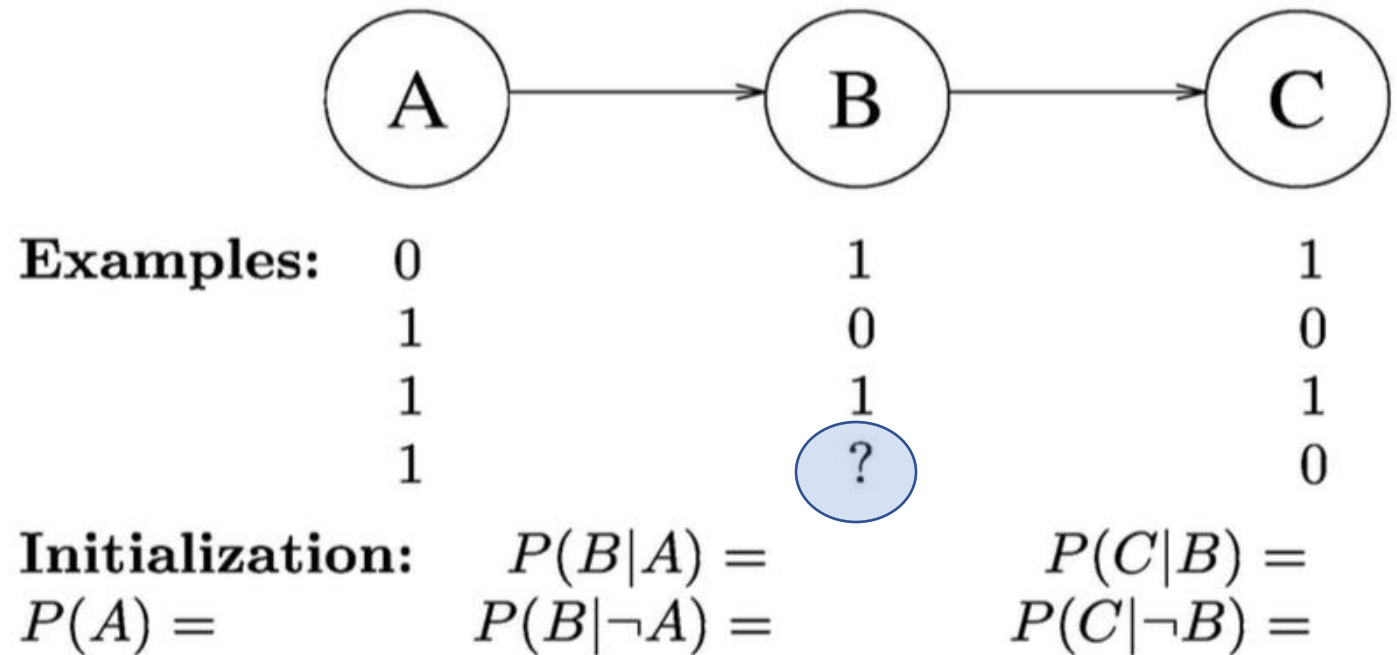
# How to learn the structure of the Bayes Net?

- Problem: Estimate/learn the structure of the model
- Setup a search process (like local search, hill climbing etc.)
- For each structure, learn the parameters.
- How to score a solution?
  - Use Max. likelihood estimation.
  - Penalize complexity of the structure (don't want a fully connected model).
  - Additionally check for validity of the conditional independences.



# Parameter Learning when some variables are not observed

- If we knew the missing value for B. Then we can estimate the CPTs.
- If we knew the CPTs then we can infer the probability of the missing value of B.
- It is a *chicken and egg* problem.

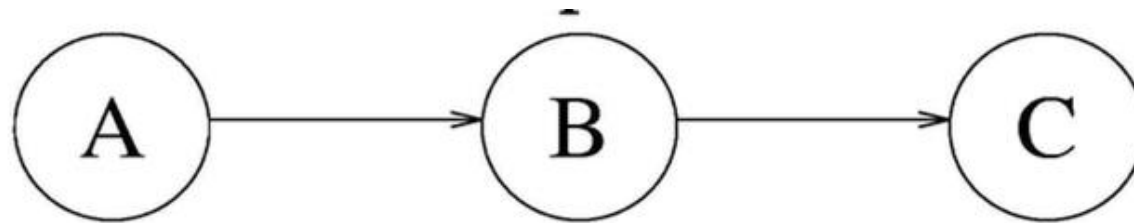


Data is incomplete. One sample has (A = 1, B = ? and C = 0)

# Expectation Maximization

- Initialization
  - Initialize CPT parameter values (ignoring missing information)
- Expectation
  - Compute expected values of unobserved variables assuming current parameters values.
  - Involves BayesNet inference (exact or approximate)
- Maximization
  - Compute new parameters (of the CPTs) to maximize the probability of data (observed and estimated)
- Alternate the EM steps until convergence. Convergence is guaranteed.

# Expectation Maximization



<b>Examples:</b>	0	1	1
	1	0	0
	1	1	1
	1	?	0

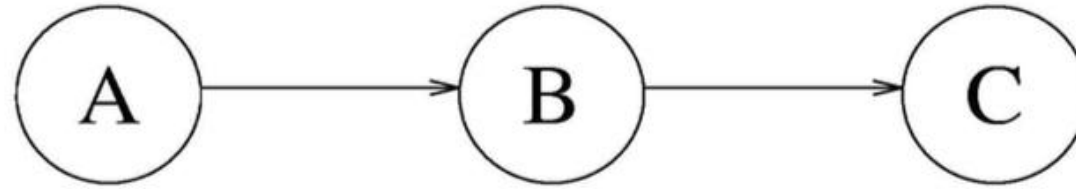
**Initialization:**  $P(B|A) = 0$   $P(C|B) = 0$   
 $P(A) = 0.75$   $P(B|\neg A) = 0$   $P(C|\neg B) = 0$

**E-step:**  $P(? = 1) = P(B|A, \neg C) = \frac{P(A, B, \neg C)}{P(A, \neg C)} = \dots = 0$

**M-step:**  $P(B|A) =$   $P(C|B) =$   
 $P(A) =$   $P(B|\neg A) =$   $P(C|\neg B) =$

**E-step:**  $P(? = 1) =$

# Expectation Maximization



<b>Examples:</b>	0	1	1
	1	0	0
	1	1	1
	1	0	0

**Initialization:**  $P(B|A) = 0$   $P(C|B) = 0$   
 $P(A) = 0.75$   $P(B|\neg A) = 0$   $P(C|\neg B) = 0$

**E-step:**  $P(? = 1) = P(B|A, \neg C) = \frac{P(A, B, \neg C)}{P(A, \neg C)} = \dots = 0$

**M-step:**  $P(B|A) = 0.33$   $P(C|B) = 1$   
 $P(A) = 0.75$   $P(B|\neg A) = 1$   $P(C|\neg B) = 0$

**E-step:**  $P(? = 1) =$

# EM Example

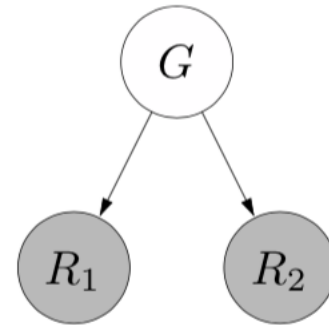
Problem: learning the parameters of a Bayes Net that models ratings given by reviewers.

We postulate that ratings (1 or 2) are conditioned on the “genre” or “type” of the move (Comedy or Drama).

Observations, we only see the ratings given by the reviewers.

Apply EM to learn the parameters.

Reviewers rate individually (their CPTs are assumed to be the same).



What if we **don't observe** some of the variables?

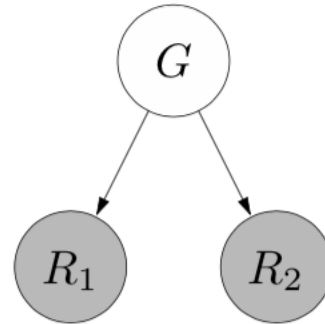
$$\mathcal{D}_{\text{train}} = \{(\textcolor{red}{?}, 4, 5), (\textcolor{red}{?}, 4, 4), (\textcolor{red}{?}, 5, 3), (\textcolor{red}{?}, 1, 2), (\textcolor{red}{?}, 5, 4)\}$$

# What objective are we optimizing in EM?

## *Maximum Marginal Likelihood*

Variables:  $H$  is hidden,  $E = e$  is observed

Example:



$H = G \quad E = (R_1, R_2) \quad e = (1, 2)$   
 $\theta = (p_G, p_R)$

Maximum marginal likelihood objective:

Marginalize over the  
latent variables in the  
likelihood

$$\begin{aligned} & \max_{\theta} \prod_{e \in \mathcal{D}_{\text{train}}} \mathbb{P}(E = e; \theta) \\ &= \max_{\theta} \prod_{e \in \mathcal{D}_{\text{train}}} \sum_h \mathbb{P}(H = h, E = e; \theta) \end{aligned}$$

# E and M steps

Initialize  $\theta$

E-step:

- Compute  $q(h) = \mathbb{P}(H = h \mid E = e; \theta)$  for each  $h$  (use any probabilistic inference algorithm)
- Create weighted points:  $(h, e)$  with weight  $q(h)$

M-step:

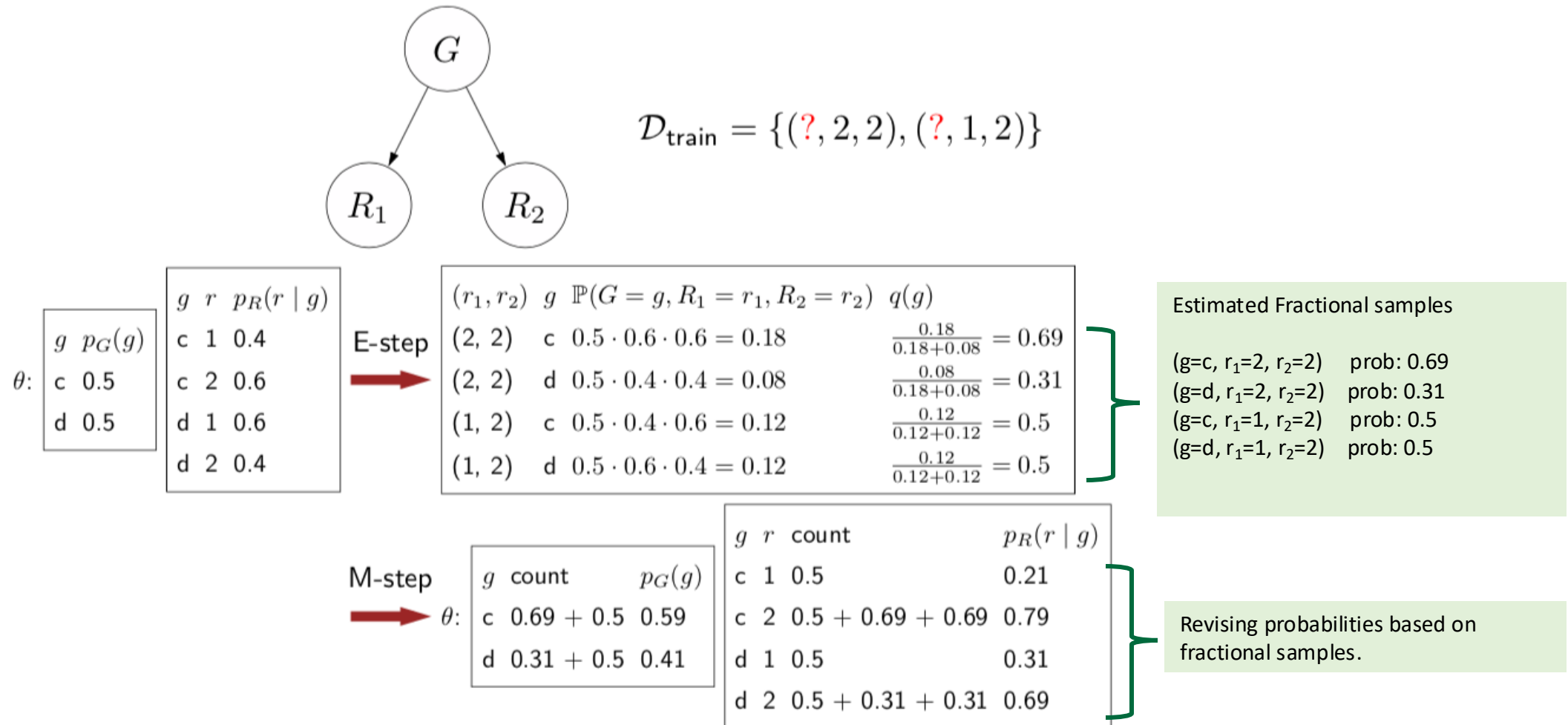
- Compute maximum likelihood (just count and normalize) to get  $\theta$

Repeat until convergence.

Compute for every value of  $h$  and for each setting of the evidence variables.

The estimated data points from E step are used to update the CPTs.

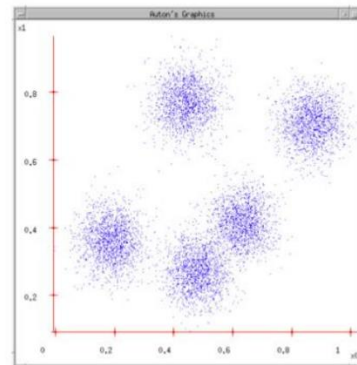
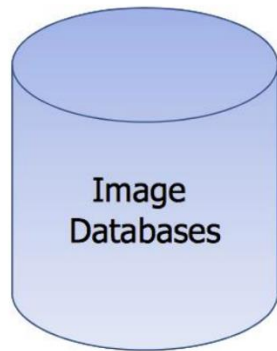
# EM: Estimating and using weighted samples



The CPTs for the two reviewers is the same.

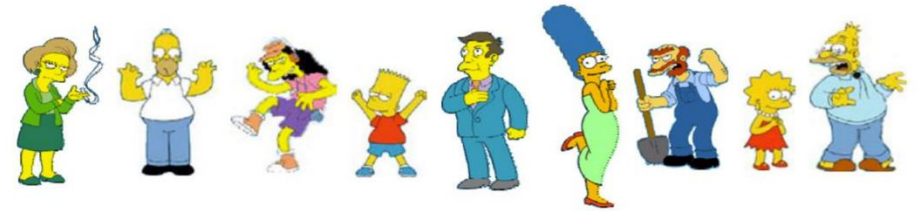
# Related Topic: Clustering

Example: Clustering images in a data base

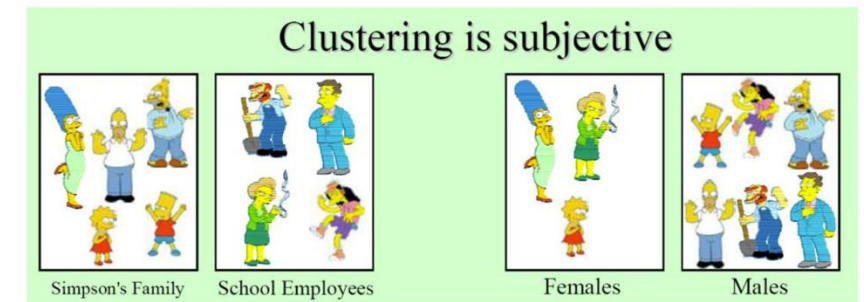


**Goal of clustering:**  
Divide object into groups,  
and objects within a group  
are more similar than  
those outside the group

Clustering is subjective



What is consider similar/dissimilar?



# Clustering is based on a distance metric

- Desired properties of dissimilarity function
  - Symmetry:  $d(x, y) = d(y, x)$ 
    - Otherwise you could claim "Alex looks like Bob, but Bob looks nothing like Alex"
  - Positive separability:  $d(x, y) = 0$ , if and only if  $x = y$ 
    - Otherwise there are objects that are different, but you cannot tell apart
  - Triangular inequality:  $d(x, y) \leq d(x, z) + d(z, y)$ 
    - Otherwise you could claim "Alex is very like Bob, and Alex is very like Carl, but Bob is very unlike Carl"

Clustering depends on the distance function used.

Euclidean distance? Edit distance? ....

# K-Means Clustering

## K-means clustering problem:

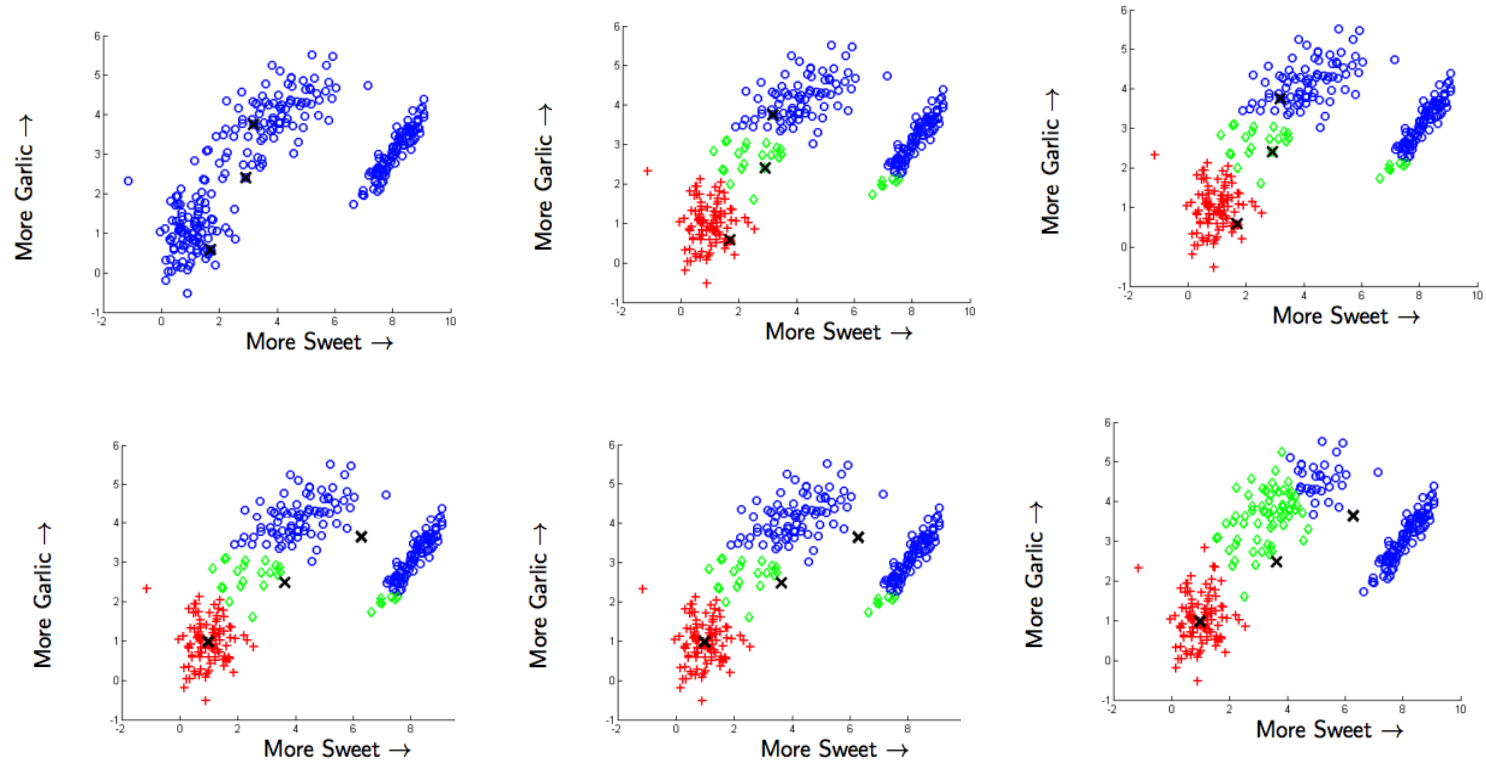
Partition the  $n$  observations into  $K$  sets ( $K \leq n$ )  $\mathbf{S} = \{S_1, S_2, \dots, S_K\}$  such that the sets minimize the within-cluster sum of squares:

$$\arg \min_{\mathbf{S}} \sum_{i=1}^K \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

where  $\boldsymbol{\mu}_i$  is the mean of points in set  $S_i$ .

A GMM yields a probability distribution over the cluster assignment for each point; whereas K-Means gives a single hard assignment

- How many different sauces should the company make?
- How sweet/garlicy should these sauces be?
- Idea: We will segment the consumers into groups (in this case 3), we will then find the best sauce for each group



# K-Means Clustering Algorithm

- Initialize  $k$  cluster centers,  $\{c^1, c^2, \dots, c^k\}$ , randomly
- Do
  - Decide the cluster memberships of each data point,  $x^i$ , by assigning it to the nearest cluster center (**cluster assignment**)

$$\pi(i) = \operatorname{argmin}_{j=1,\dots,k} \|x^i - c^j\|^2$$

- Adjust the cluster centers (**center adjustment**)

$$c^j = \frac{1}{|\{i: \pi(i) = j\}|} \sum_{i: \pi(i)=j} x^i$$

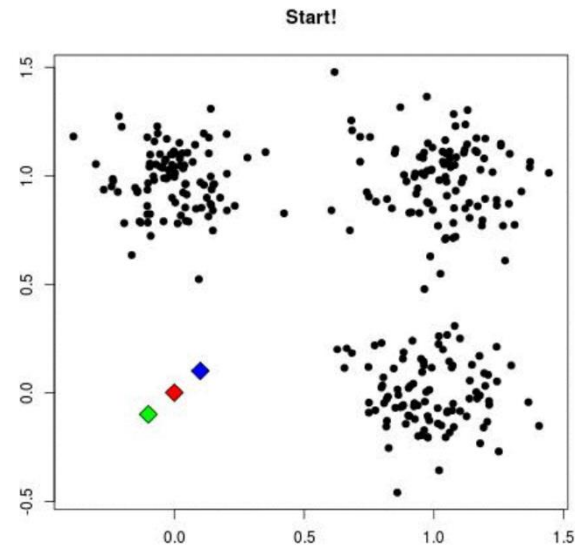
- While any cluster center has been changed

# What objective K-Means is optimizing?

- Given  $n$  data points,  $\{x^1, x^2, \dots, x^n\} \in R^d$
- Find  $k$  cluster centers,  $\{c^1, c^2, \dots, c^k\} \in R^d$
- And assign each data point  $i$  to one cluster,  $\pi(i) \in \{1, \dots, k\}$
- Such that the averaged square distance from each data point to respective cluster center (distortion metric) is minimum:

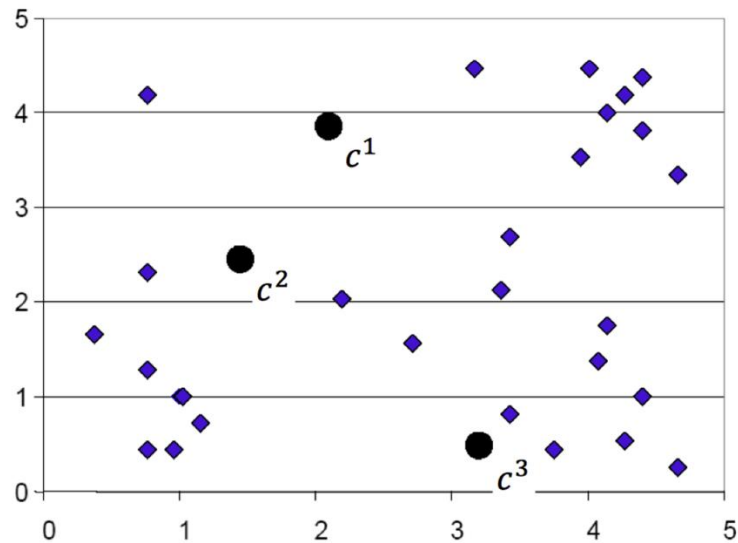
$$\min_{c, \pi} \frac{1}{n} \sum_{i=1}^n \|x^i - c^{\pi(i)}\|^2$$

Data points

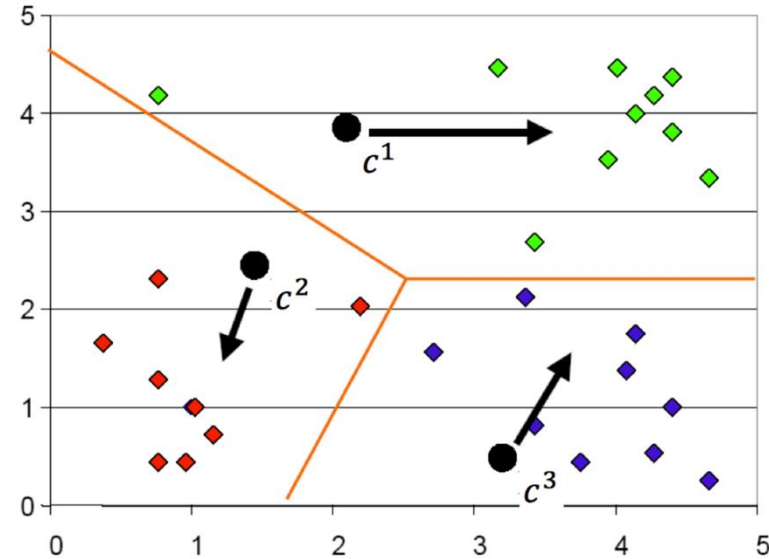


K-means converges with every step to the minimum distortion metric

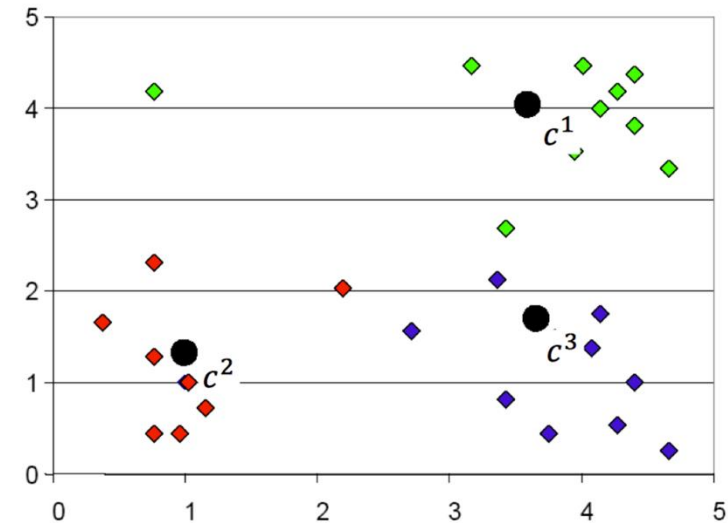
Iteration I



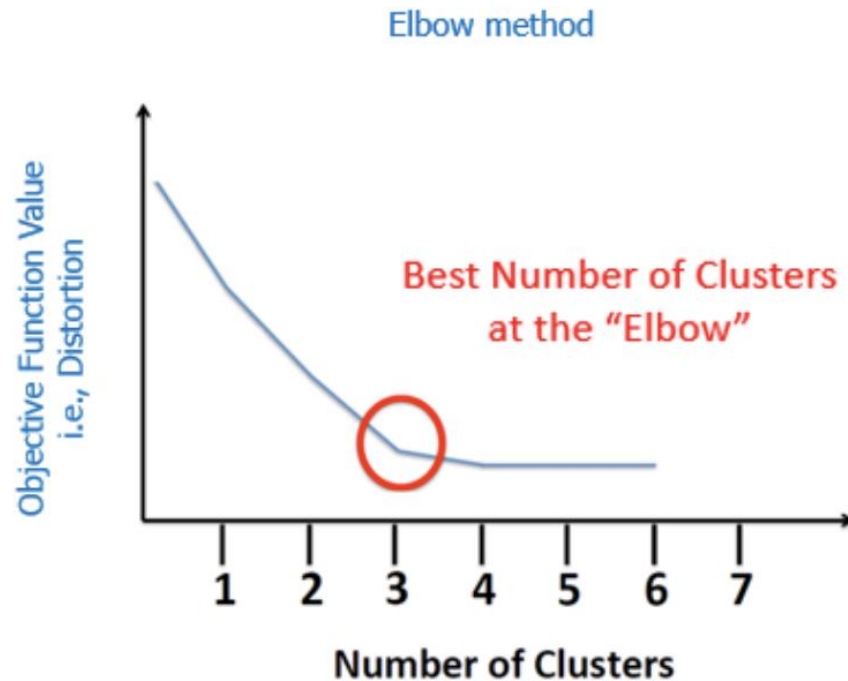
Iteration II



Iteration III



# How to pick “k”?



Ideal value of number of clusters(k) can be identified using the distortion metric for different values of k.

**Distortion score:** computing the sum of squared distances from each point to its assigned center

# K-Means Application: Segmentation

Goal of segmentation is to partition an image into regions each of which has reasonably homogenous visual appearance.

Apply K-Means in the colour space.

K=2



K=3



K=10



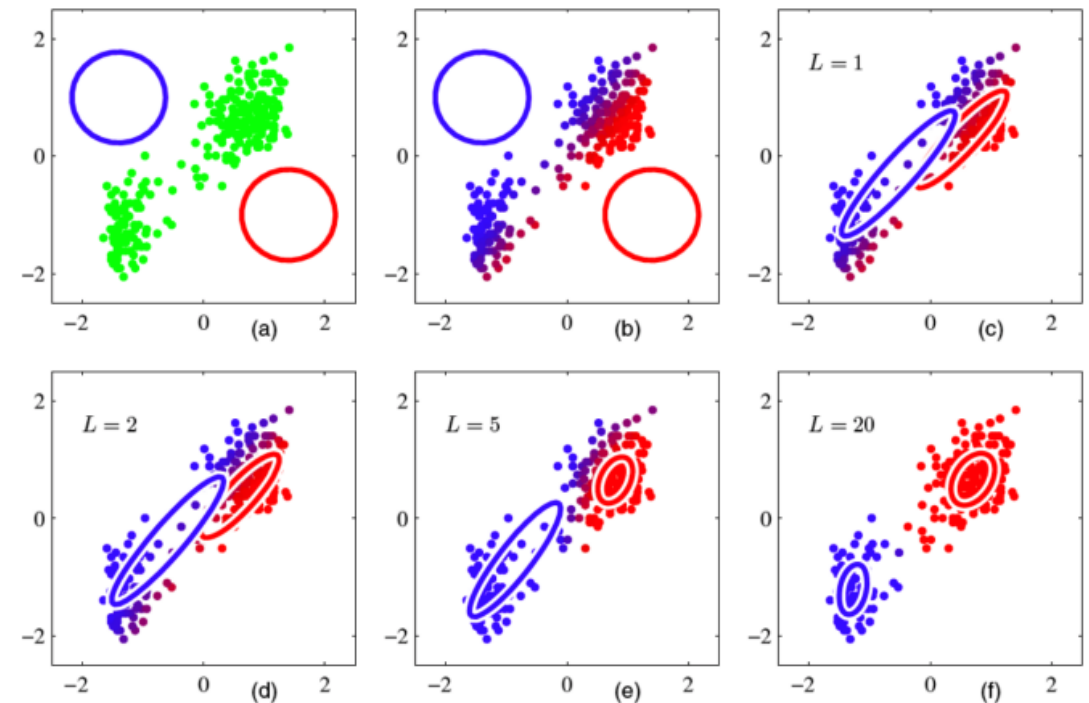
Original



# EM in Continuous Space: Gaussian Mixture Modeling

- Problem: clustering task where we want to discern multiple category in a collection of given points.
- Assume a mixture of components (Gaussian)
- Don't know which data point comes from which component.
- Use EM to iteratively determine the assignments and the parameters of the Gaussian components.

$$P(x) = \sum_{i=1}^k P(C = i)P(x|C = i)$$



# Soft vs. hard assignments during clustering

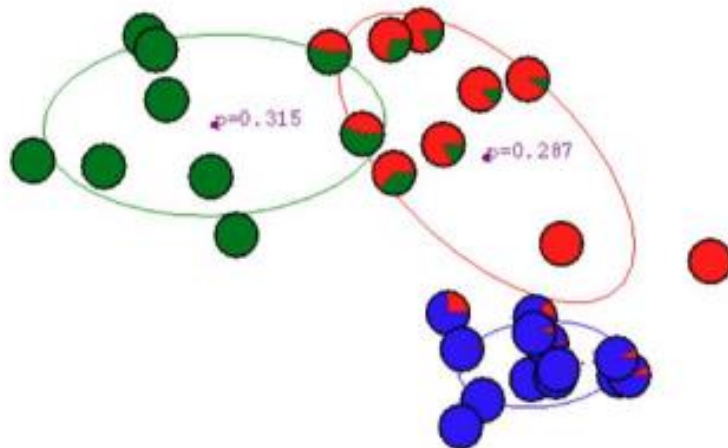
- **K-means**

- hard assignment:** each object belongs to only one cluster

$$\theta_i \in \{\theta_1, \dots, \theta_K\}$$

- **Mixture modeling**

- soft assignment:** probability that an object belongs to a cluster



# Gaussian Mixture Models (GMMs)

- Formally a Mixture Model is the weighted sum of a number of pdfs where the weights are determined by a distribution,  $\pi$

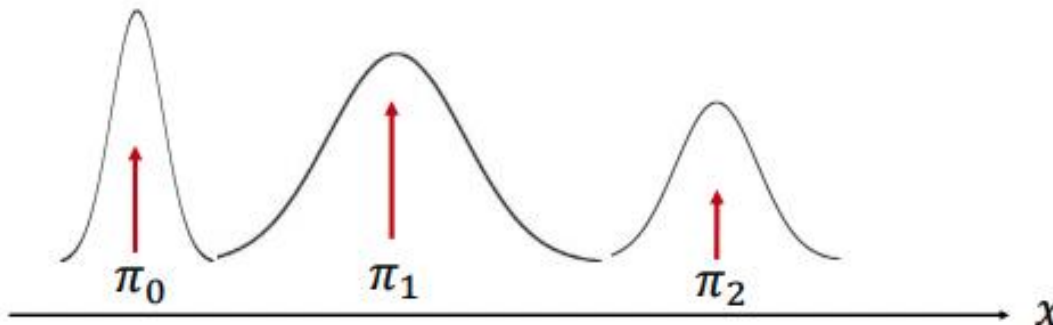
$$p(x) = \pi_0 f_0(x) + \pi_1 f_1(x) + \pi_2 f_2(x) + \dots + \pi_k f_k(x)$$

where  $\sum_{i=0}^k \pi_i = 1$

$$p(x) = \sum_{i=0}^k \pi_i f_i(x)$$

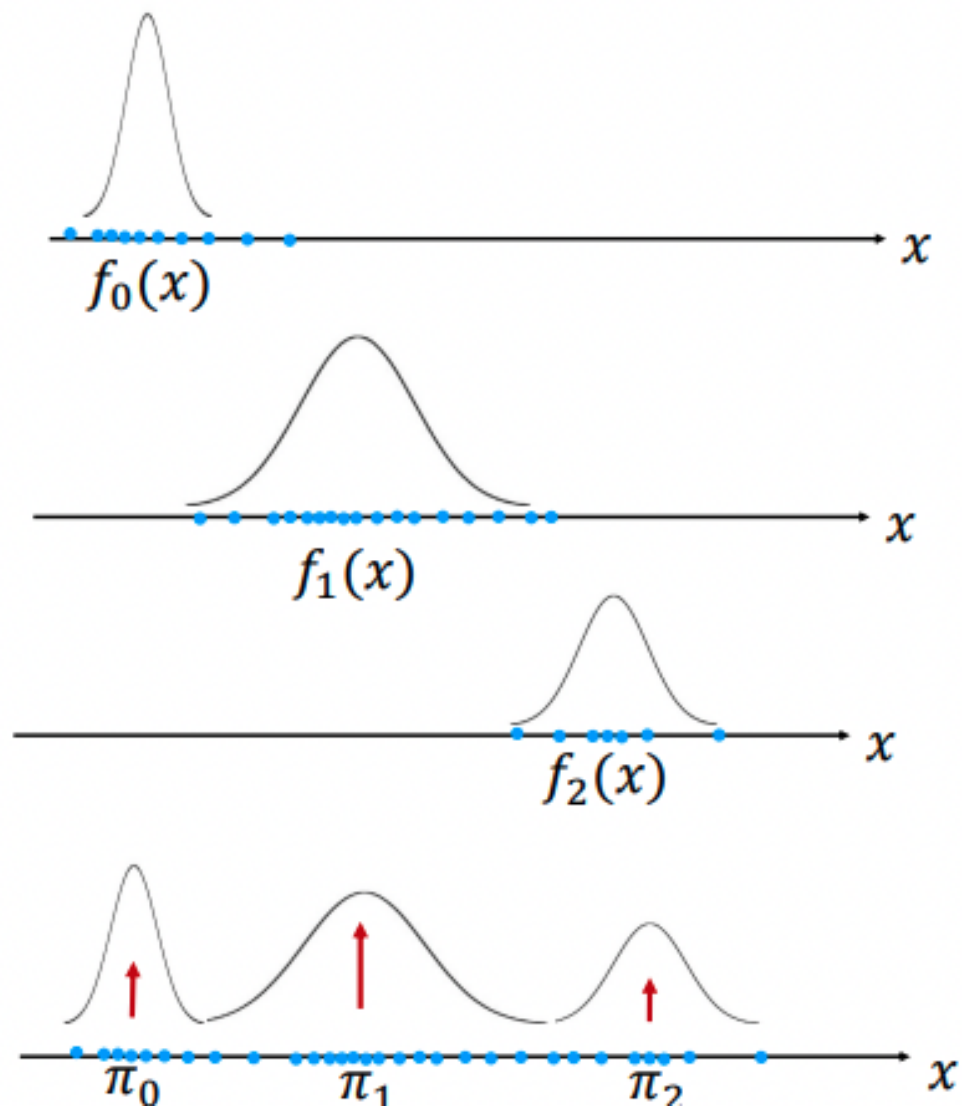
GMMs are a generative model of data.

They model how the data was generated from an underlying model.

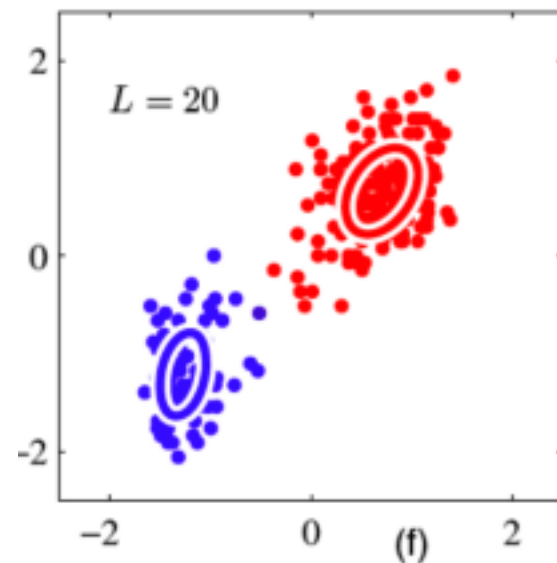


What is **f** in GMM?

$$p(x) = \pi_0 f_0(x) + \pi_1 f_1(x) + \pi_2 f_2(x)$$



Each  $f$  is the normal distribution. The overall data set is generated as being sampled from a mixture.



# Learning a GMM: Optimizing the likelihood of generating the data

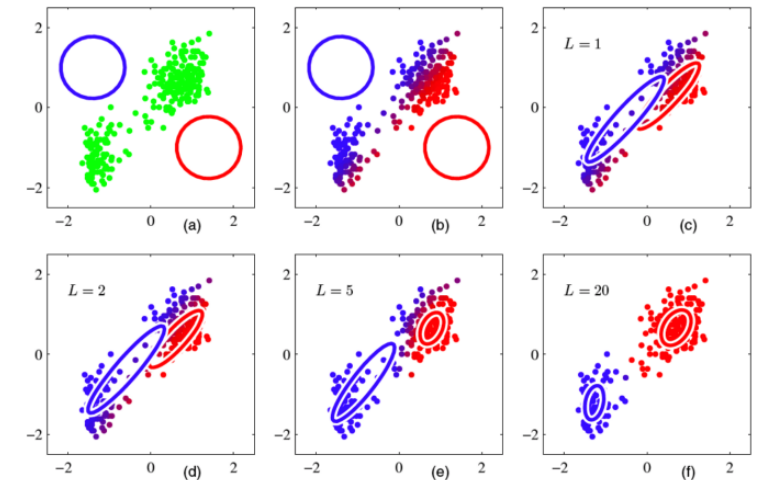
$$\arg \max p(x) = p(x|\pi, \mu, \Sigma) = \prod_{n=1}^N p(x_n|\theta) = \prod_{n=1}^N \sum_{k=0}^K \pi_k N(x_n|\mu_k, \Sigma_k)$$

$$\ln[p(x)] = \ln[p(x|\pi, \mu, \Sigma)]$$

- As usual: Identify a likelihood function

$$\ln p(x|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k N(x_n|\mu_k, \Sigma_k) \right\}$$

- And set partials to zero...



We want to fit the parameters of the Gaussian mixture model (mixing fractions and the parameters of the Gaussians given the data).

# E-step (associating data points with clusters)

- The conditional of  $p(z_{nk}|x, \theta)$  can be derived using Bayes rule.
  - The **responsibility** that a mixture component takes for explaining an observation  $x$ .

$$\begin{aligned}\tau(z_k) = p(z_k = 1|x) &= \frac{p(z_k = 1)p(x|z_k = 1)}{\sum_{j=1}^K p(z_j = 1)p(x|z_j = 1)} \\ &= \frac{\pi_k N(x|\mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j N(x|\mu_j, \Sigma_j)}\end{aligned}$$

# M-step (given responsibilities optimize the GMM parameters)

- Optimization of means.

$$\begin{aligned}\ln p(x|\pi, \mu, \Sigma) &= \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k N(x_n | \mu_k, \Sigma_k) \right\} \\ \frac{\partial \ln p(x|\pi, \mu, \Sigma)}{\partial \mu_k} &= \sum_{n=1}^N \frac{\pi_k N(x_n | \mu_k, \Sigma_k)}{\sum_j \pi_j N(x_n | \mu_j, \Sigma_j)} \Sigma_k^{-1} (x_n - \mu_k) = 0 \\ &= \sum_{n=1}^N \tau(z_{nk}) \Sigma_k^{-1} (x_n - \mu_k) = 0 \\ \mu_k &= \frac{\sum_{n=1}^N \tau(z_{nk}) x_n}{\sum_{n=1}^N \tau(z_{nk})}\end{aligned}$$

- Optimization of covariance

$$\ln p(x|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k N(x_n | \mu_k, \Sigma_k) \right\}$$

$$\Sigma_k = \frac{1}{\sum_{n=1}^N \tau(z_{nk})} \sum_{n=1}^N \tau(z_{nk}) (x_n - \mu_k)(x_n - \mu_k)^T$$

# EM for GMMs

E-step: Evaluate the Responsibilities

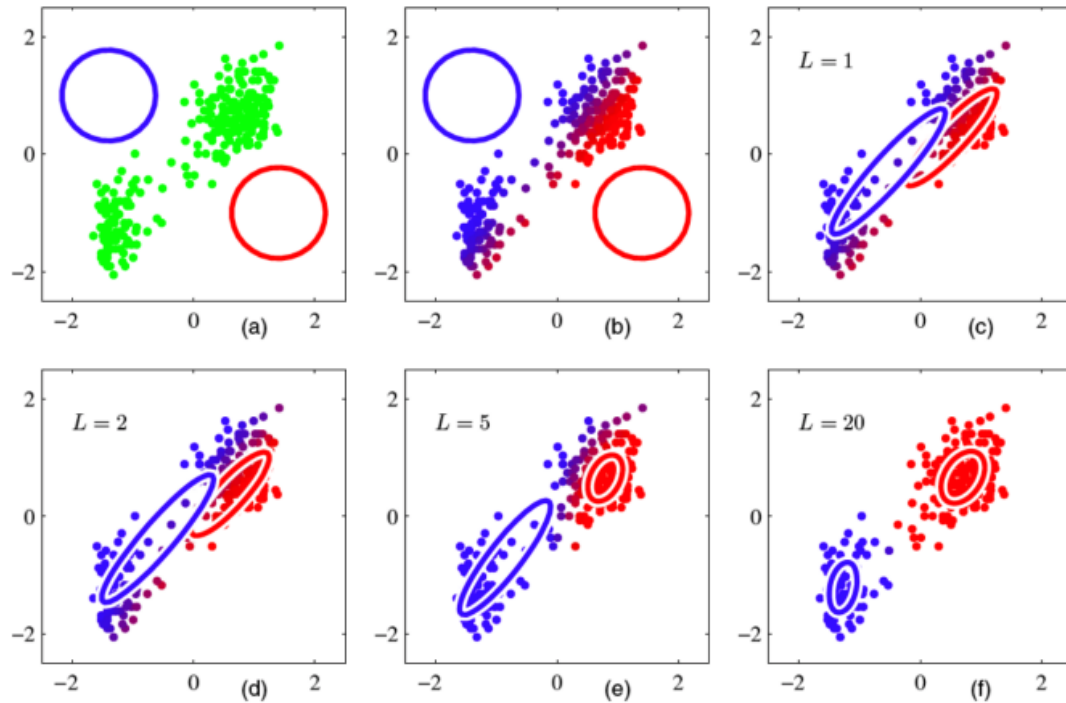
$$\tau(z_{nk}) = \frac{\pi_k N(x_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j N(x_n | \mu_j, \Sigma_j)}$$

M-Step: Re-estimate Parameters

$$\mu_k^{new} = \frac{\sum_{n=1}^N \tau(z_{nk}) x_n}{N_k}$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^N \tau(z_{nk}) (x_n - \mu_k^{new})(x_n - \mu_k^{new})^T$$

$$\pi_k^{new} = \frac{N_k}{N}$$

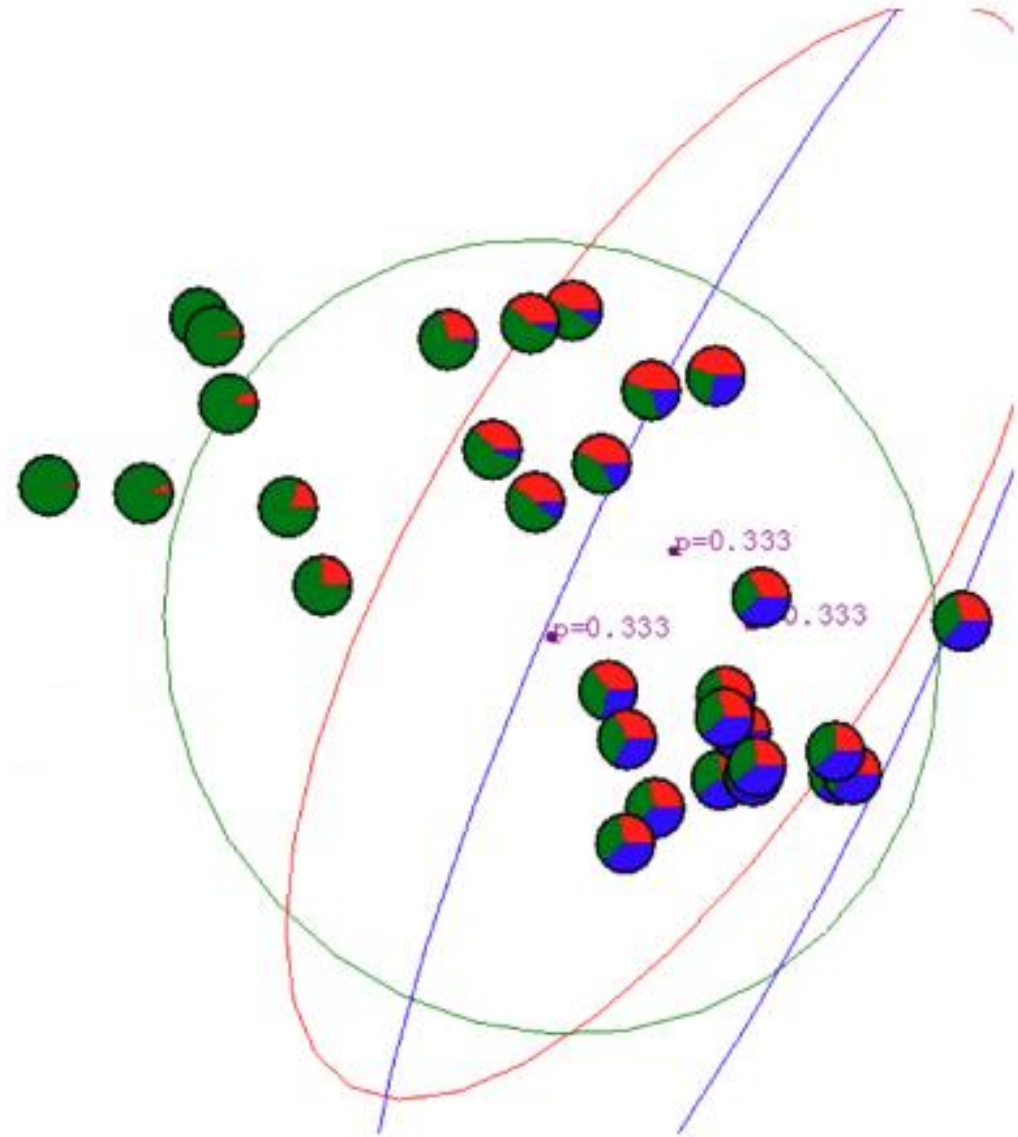


# GMM Example

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

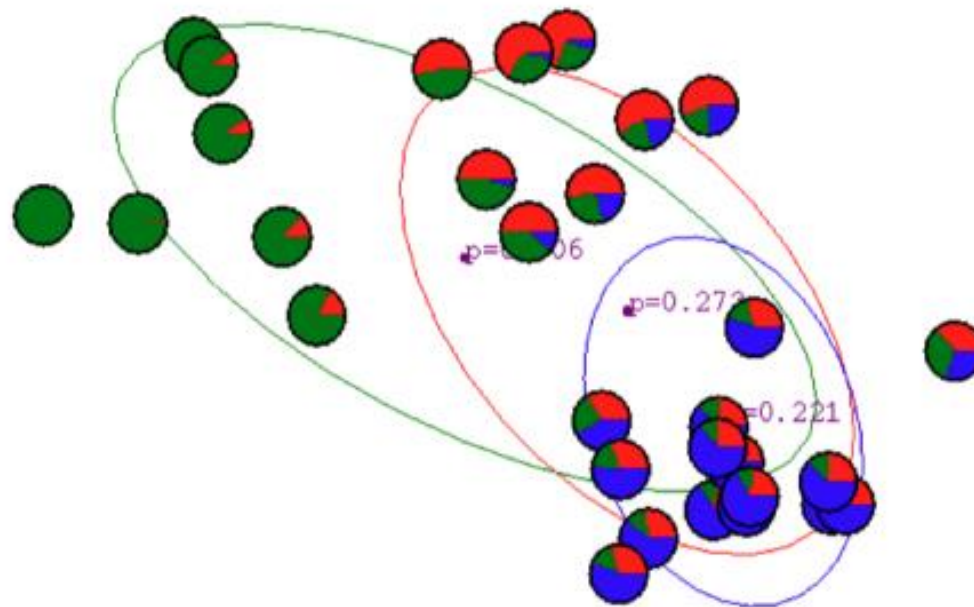


Colours indicate cluster membership likelihood



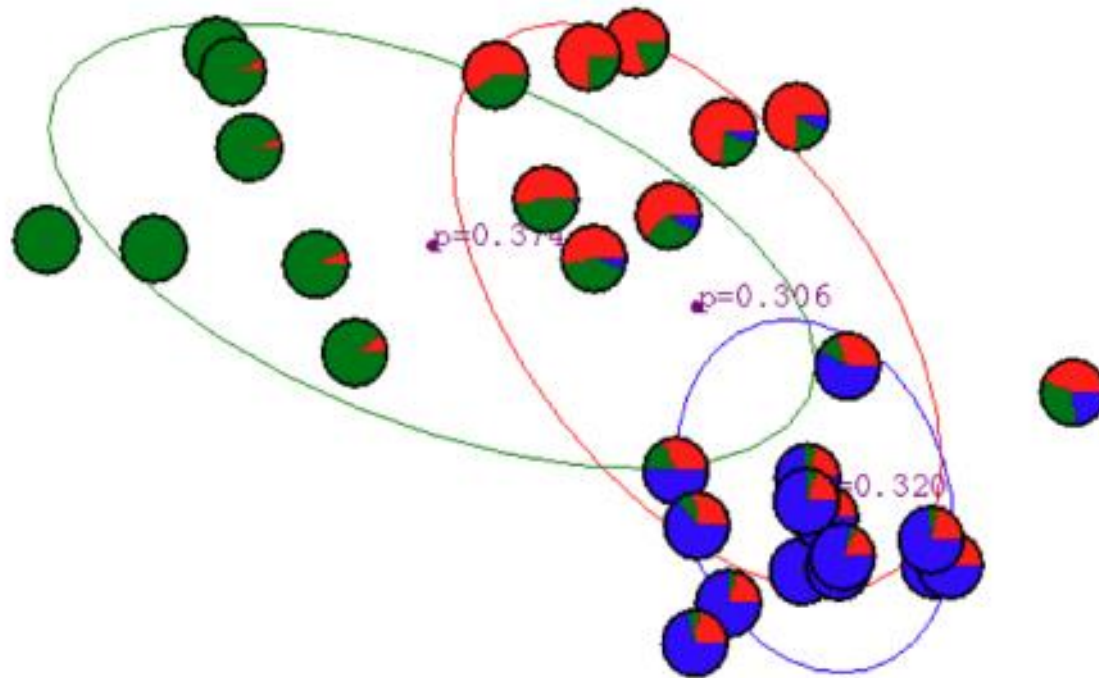
# Example

After 1<sup>st</sup> iteration



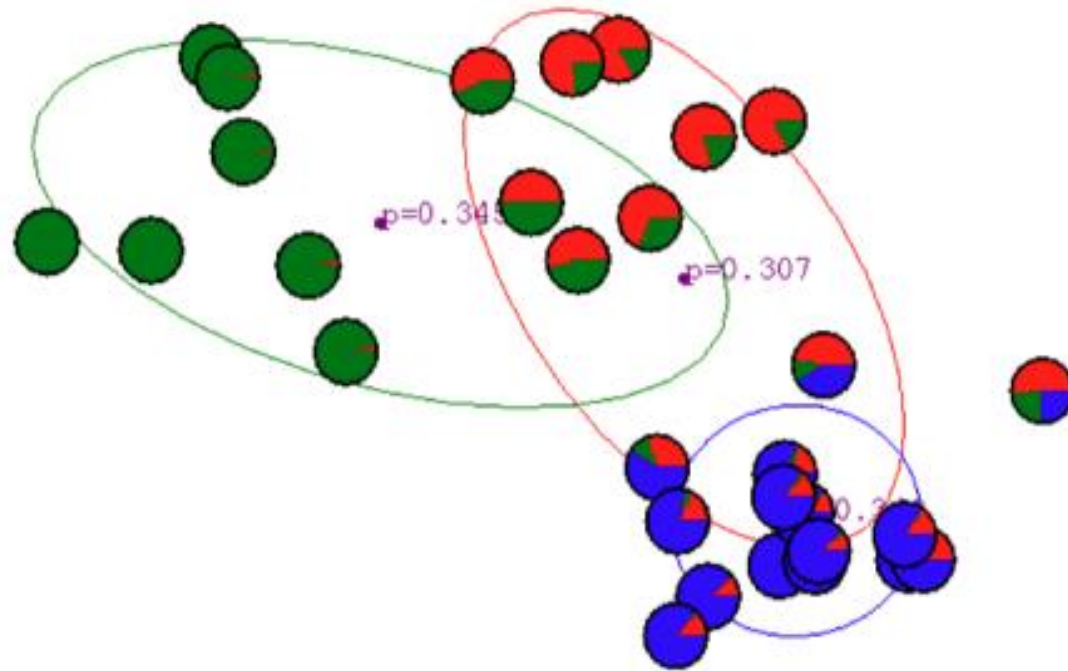
# Example

After 2<sup>nd</sup> iteration



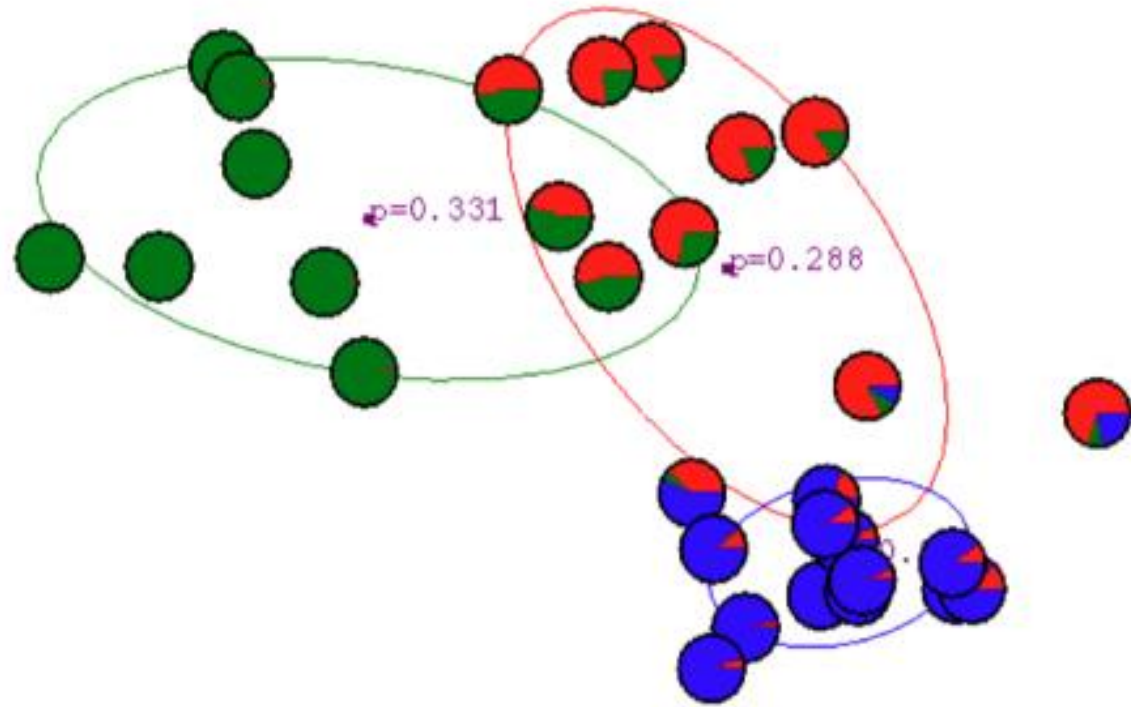
# Example

After 3<sup>rd</sup> iteration



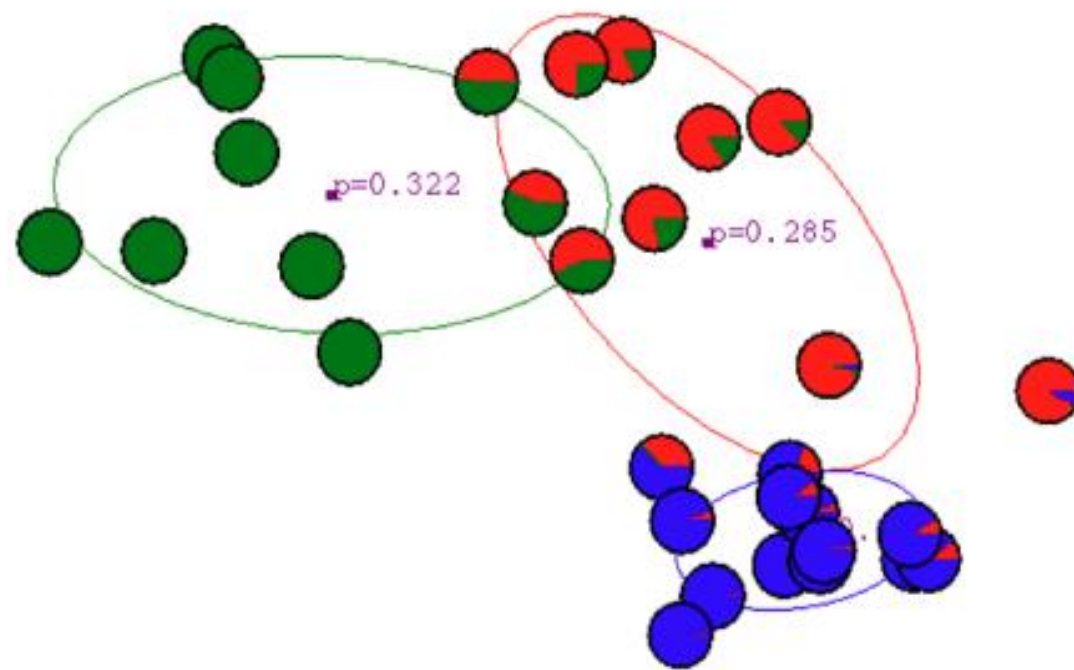
# Example

After 4<sup>th</sup> iteration



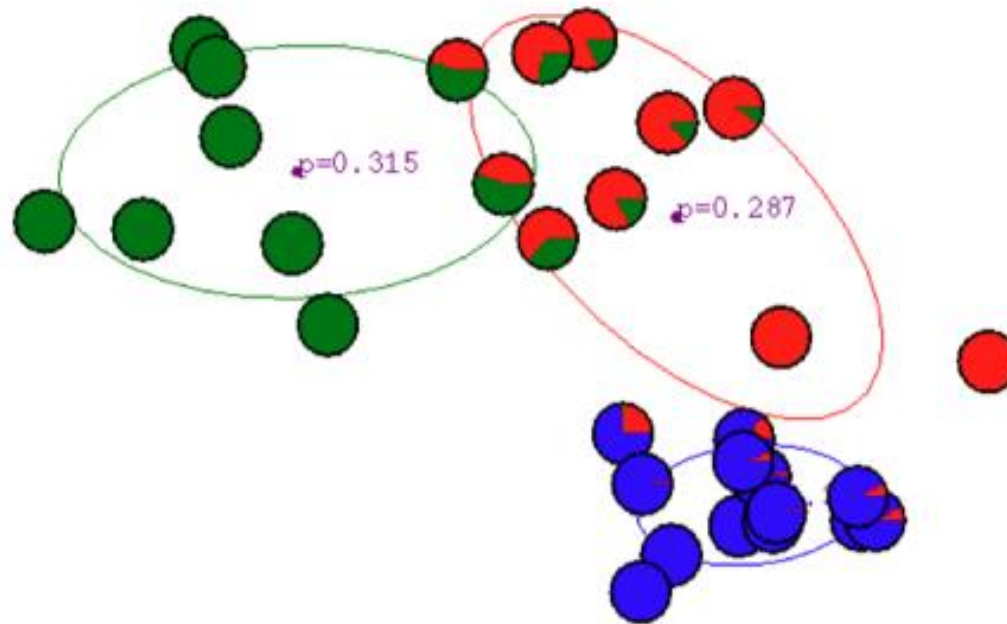
# Example

After 5<sup>th</sup> iteration



# Example

After 6<sup>th</sup> iteration



# Example

**After 20<sup>th</sup> iteration**

