Machine Learning

Expectation Maximization and Gaussian Mixtures

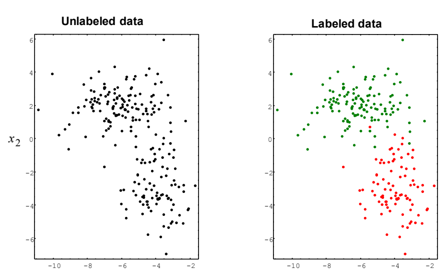
CSE 473
Chapter 20.3

Feedback in Learning

- Supervised learning: correct answers for each example
- Unsupervised learning: correct answers not given
- Reinforcement learning: occasional rewards

The problem of finding labels for unlabeled data

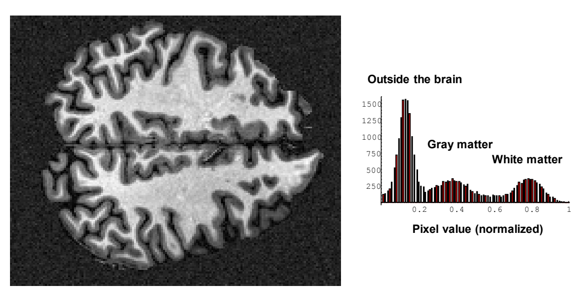
So far we have solved "supervised" classification problems where a teacher told us the label of each example. In nature, items often do not come with labels. How can we learn labels without a teacher?



From Shadmehr & Diedrichsen

Example: image segmentation

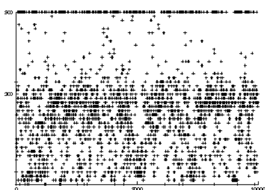
Identify pixels that are white matter, gray matter, or outside of the brain.



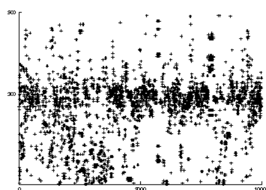
From Shadmehr & Diedrichsen

Raw Proximity Sensor Data

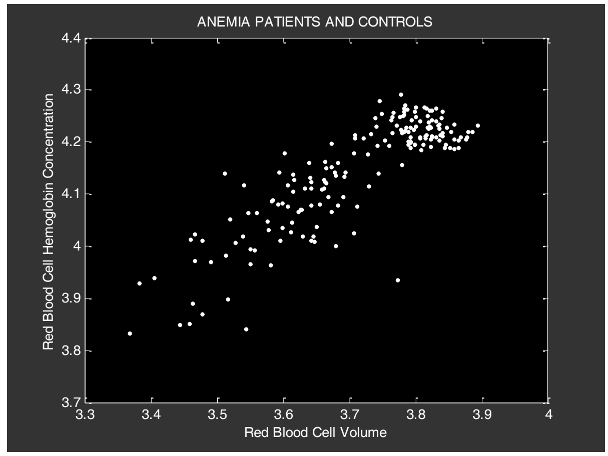
Measured distances for expected distance of 300 cm.



Sonar



Laser

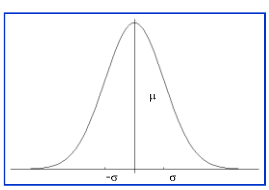


Gaussians

$p(x) \sim N(\mu, \sigma^2):$

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

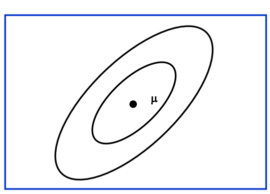
Univariate



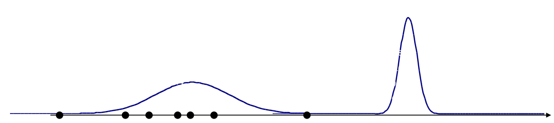
$p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma}):$

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

Multivariate



Fitting a Gaussian PDF to Data



From Russell

Fitting a Gaussian PDF to Data

- Suppose $y = y_1,\ldots,y_n,\ldots,y_N$ is a set of N data values
- Given a Gaussian PDF p with mean μ and variance σ, define:

$$p(y \mid \mu,\sigma) = \prod_{n=1}^{N} p(y_n \mid \mu,\sigma) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{1}{2}\frac{(y_n-\mu)^2}{\sigma^2}}$$

- How do we choose μ and σ to maximise this probability?

From Russell

Maximum Likelihood Estimation

- Define the best fitting Gaussian to be the one such that $p(y|\mu,\sigma)$ is maximised.
- Terminology:
  - $p(y|\mu,\sigma)$, thought of as a function of y is the probability (density) of y
  - $p(y|\mu,\sigma)$, thought of as a function of μ,σ is the likelihood of μ,σ
- Maximizing $p(y|\mu,\sigma)$ with respect to μ,σ is called Maximum Likelihood (ML) estimation of μ,σ

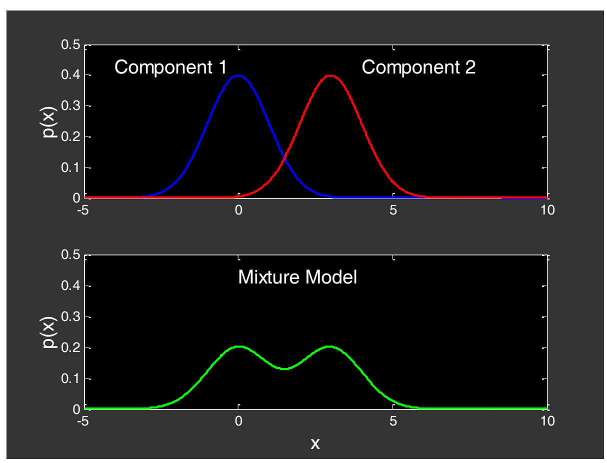
From Russell

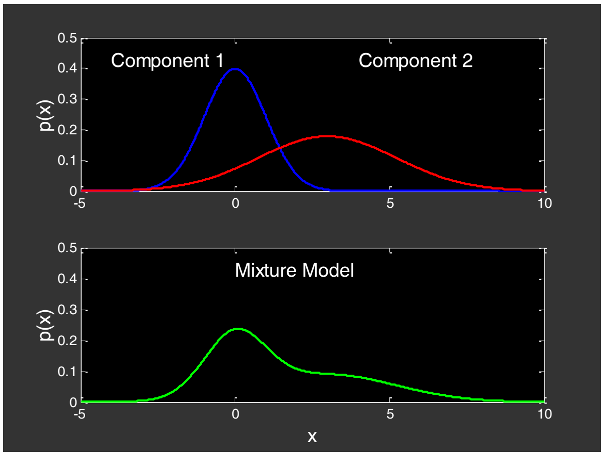
ML estimation of μ,σ

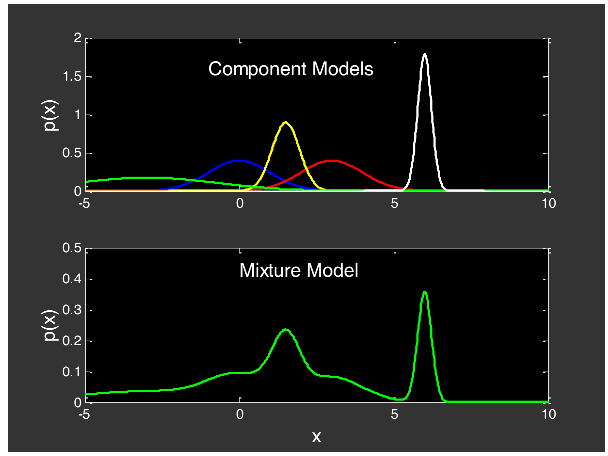
- Intuitively:
  - The maximum likelihood estimate of μ should be the average value of $y_1,\ldots,y_N$, (the sample mean)
  - The maximum likelihood estimate of σ should be the variance of $y_1,\ldots,y_N$ (the sample variance)
- This turns out to be true: $p(y|\ \mu,\ \sigma)$ is maximised by setting:

$$\mu = \frac{1}{N}\sum_{n=1}^{N} y_n, \qquad \sigma = \frac{1}{N}\sum_{n=1}^{N}(y_n-\mu)^2$$

From Russell







Mixtures

If our data is not labeled, we can hypothesize that:

1. There are exactly m classes in the data: $y \in \{1, 2, \cdots, m\}$
2. Each class y occurs with a specific frequency: $P(y)$
3. Examples of class y are governed by a specific distribution: $p(\mathbf{x} \mid y)$

According to our hypothesis, each example $\mathbf{x}^{(i)}$ must have been generated from a specific "mixture" distribution:

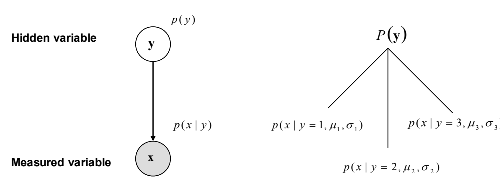
$$p(\mathbf{x}) = \sum_{j=1}^{m} P(y=j)\, p(\mathbf{x} \mid y=j)$$

We might hypothesize that the distributions are Gaussian:

Parameters of the distributions $\theta = \{P(y=1), \mu_1, \Sigma_1, \cdots, P(y=m), \mu_m, \Sigma_m\}$

$$p(\mathbf{x} \mid \theta) = \sum_{j=1}^{m} \underbrace{P(y=j)}_{\text{Mixing proportions}} \underbrace{N(\mathbf{x} \mid \mu_j, \Sigma_j)}_{\text{Normal distribution}}$$

Graphical Representation of Gaussian Mixtures



$$p(x) = \sum_{i=1}^{3} p(y=i)\, p(x \mid y=i, \mu_i, \sigma_i)$$

Learning of mixture models

Learning Mixtures from Data

Consider fixed K = 2

e.g., Unknown parameters $\Theta = \{\mu_1 , \sigma_1 , \mu_2 , \sigma_2 , \alpha_1\}$

Given data $D = \{x_1,\ldots\ldots x_N\}$, we want to find the parameters Θ that "best fit" the data

Early Attempts

Weldon's data, 1893

- n=1000 crabs from Bay of Naples
- Ratio of forehead to body length
- Suspected existence of 2 separate species

Early Attempts

Karl Pearson, 1894:

- JRSS paper
- proposed a mixture of 2 Gaussians
- 5 parameters $\Theta = \{\mu_1 , \sigma_1 , \mu_2 , \sigma_2 , \alpha_1\}$

- parameter estimation -> method of moments
- involved solution of 9th order equations!

(see Chapter 10, Stigler (1986), The History of Statistics)

"The solution of an equation of the ninth degree, where almost all powers, to the ninth, of the unknown quantity are existing, is, however, a very laborious task. Mr. Pearson has indeed possessed the energy to perform his heroic task…. But I fear he will have few successors….."

Charlier (1906)

Maximum Likelihood Principle

- Fisher, 1922
  - assume a probabilistic model
  - likelihood = p(data | parameters, model)
  - find the parameters that make the data most likely

1977: The EM Algorithm

- Dempster, Laird, and Rubin
  - General framework for likelihood-based parameter estimation with missing data
    - start with initial guesses of parameters
    - E-step: estimate memberships given params
    - M-step: estimate params given memberships
    - Repeat until convergence
  - Converges to a (local) maximum of likelihood
  - E-step and M-step are often computationally simple
  - Generalizes to maximum a posteriori (with priors)

EM for Mixture of Gaussians

- E-step: Compute probability that point x_j was generated by component i:

$$p_{ij} = P(C = i \mid x_j)$$

$$p_{ij} = \alpha\ P(x_j \mid C = i)\ P(C = i)$$

$$p_i = \sum_j p_{ij}$$

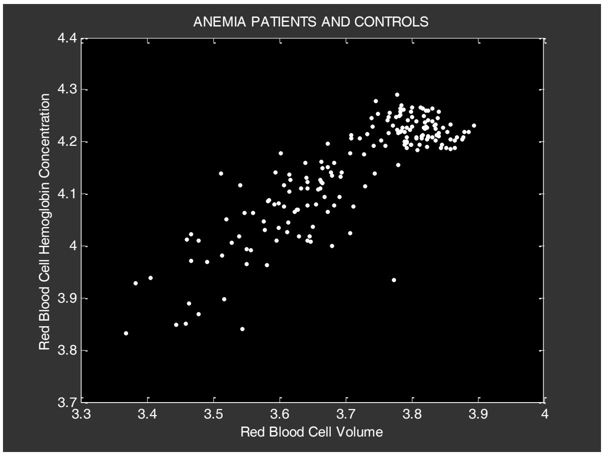
- M-step: Compute new mean, covariance, and component weights:

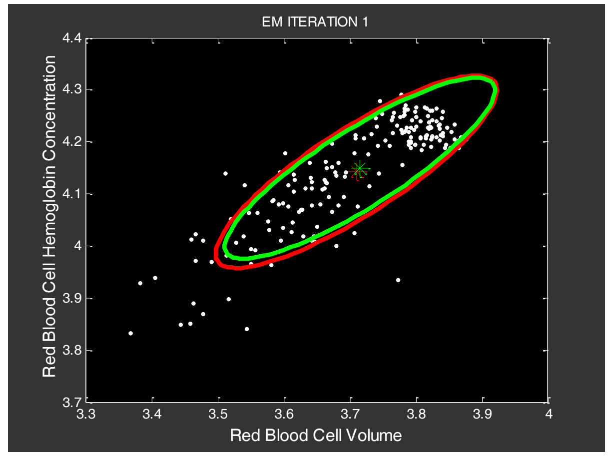
$$\mu_i \leftarrow \sum_j p_{ij} x_j / p_i$$

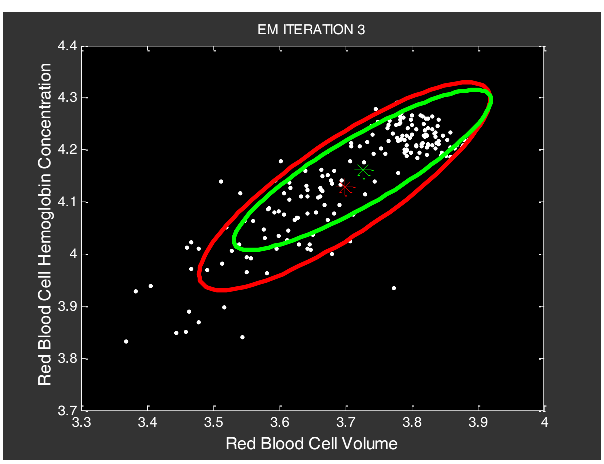
$$\sigma^2 \leftarrow \sum_j p_{ij} (x_j - \mu_i)^2 / p_i$$

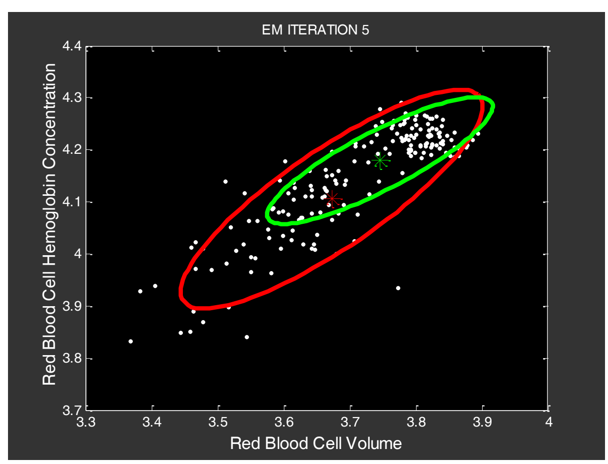
$$w_i \leftarrow p_i$$

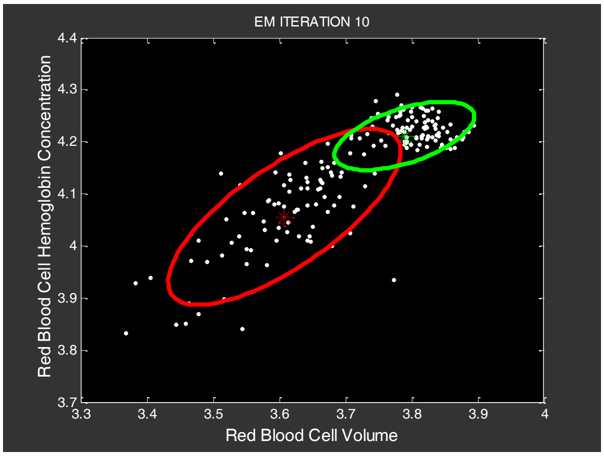
24

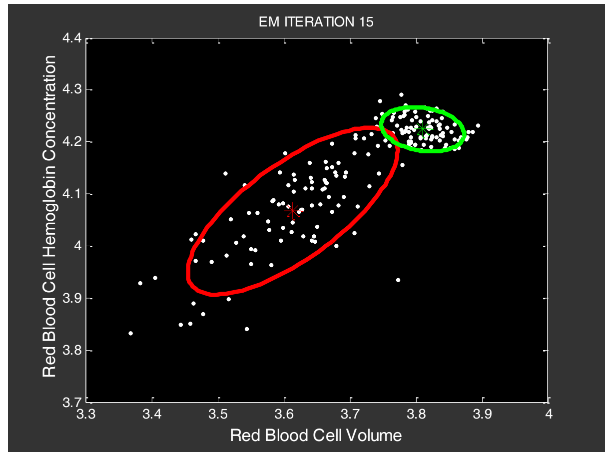

ANEMIA PATIENTS AND CONTROLS
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
3.3 3.4 3.5 3.6 3.7 3.8 3.9 4
3.7 3.8 3.9 4 4.1 4.2 4.3 4.4

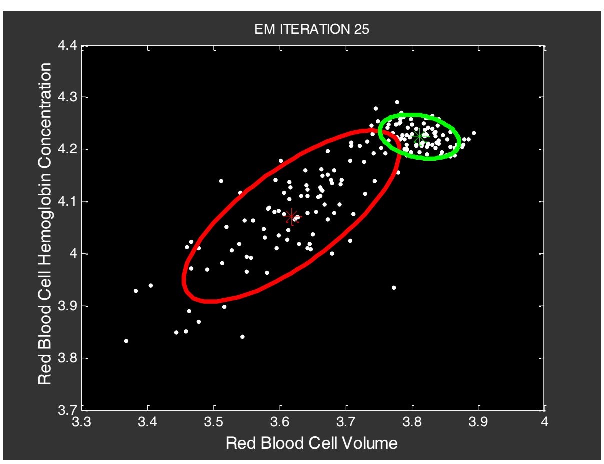

EM ITERATION 1
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
3.3 3.4 3.5 3.6 3.7 3.8 3.9 4
3.7 3.8 3.9 4 4.1 4.2 4.3 4.4

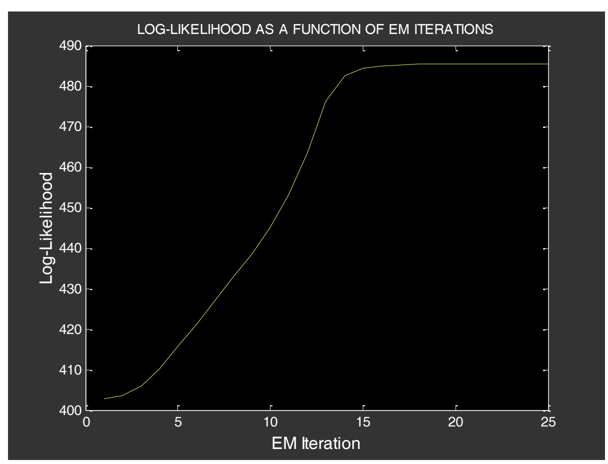

EM ITERATION 3
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
3.3 3.4 3.5 3.6 3.7 3.8 3.9 4
3.7 3.8 3.9 4 4.1 4.2 4.3 4.4

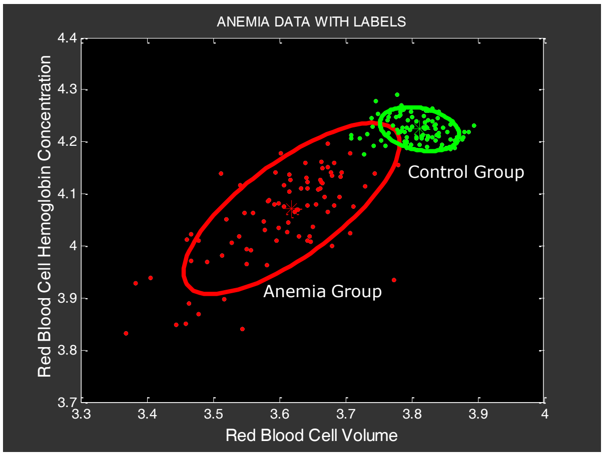

EM ITERATION 5
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
3.3 3.4 3.5 3.6 3.7 3.8 3.9 4
3.7 3.8 3.9 4 4.1 4.2 4.3 4.4


EM ITERATION 10
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4

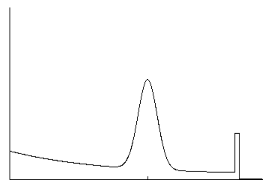

EM ITERATION 15
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4


EM ITERATION 25
Red Blood Cell Hemoglobin Concentration
Red Blood Cell Volume
4.4
4.3
4.2
4.1
4
3.9
3.8
3.7
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4


LOG-LIKELIHOOD AS A FUNCTION OF EM ITERATIONS
Log-Likelihood
EM Iteration
490
480
470
460
450
440
430
420
410
400
0
5
10
15
20
25



Mixture Density

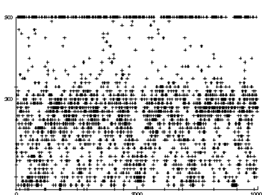


$$P(z \mid x, m) = \begin{pmatrix} \alpha_{\text{hit}} \\ \alpha_{\text{unexp}} \\ \alpha_{\max} \\ \alpha_{\text{rand}} \end{pmatrix}^T \cdot \begin{pmatrix} P_{\text{hit}}(z \mid x, m) \\ P_{\text{unexp}}(z \mid x, m) \\ P_{\max}(z \mid x, m) \\ P_{\text{rand}}(z \mid x, m) \end{pmatrix}$$

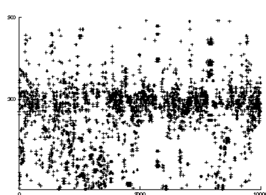
How can we determine the model parameters?

Raw Sensor Data

Measured distances for expected distance of 300 cm.

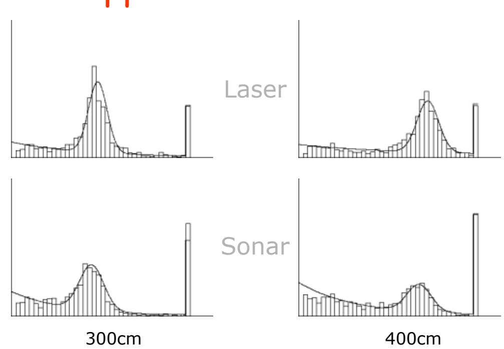


Sonar

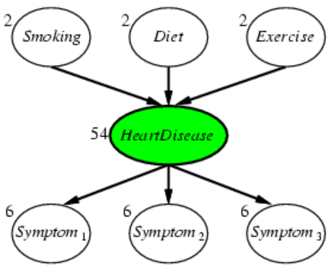


Laser

Approximation Results



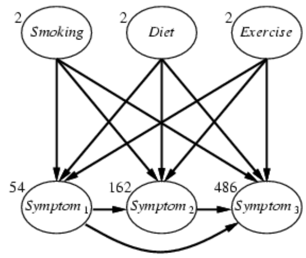
Hidden Variables



- But we can't observe the disease variable
- Can't we learn without it?

We *-could-*

- But we'd get a fully-connected network



With 708 parameters (vs. 78)

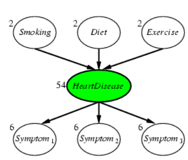
Much harder to learn!

Chicken & Egg Problem

- If we knew that a training instance (patient) had the disease…
  - It would be easy to learn P(symptom | disease)
  - But we can't observe disease, so we don't.

If we knew params, e.g. P(symptom | disease) then it'd be easy to estimate if the patient had the disease.

But we don't know these parameters.



Expectation Maximization (EM)

(high-level version)

- Pretend we ***do*** know the parameters
  - Initialize randomly
- **[E step]** Compute probability of instance having each possible value of the hidden variable

[M step] Treating each instance as fractionally having **both** values compute the new parameter values

Iterate until convergence!