COMP 4360 Machine Learning
Bayesian Learning

Jacky Baltes
Autonomous Agents Lab
University of Manitoba
Winnipeg, Canada
R3T 2N2
Email: jacky@cs.umanitoba.ca
WWW: http://www.cs.umanitoba.ca/~jacky
http://aalab.cs.umanitoba.ca

Question

- There are three doors. One door has a bag of gold. The other two doors have tigers behind them. The king asks you to select a door. If you open the door with the bag of gold behind it, you can leave with the gold. Otherwise you are lunch for the tigers. You select the left door. The king opens the middle door and shows you a caged tiger.
- The king asks you again, which door you want to choose. Should you change and pick the left door?

Monty's Dilemma

- Assume gold is behind the first door
- Strategy 1: Player picks one door and stays with it
- Door A 1/3
  - King opens Door B
    - Player keeps Door A -> Gold: 1/3 * ½
  - King opens Door C
    - Player keeps Door A -> Gold: 1/3 * ½
- Door B 1/3
  - King opens Door C
    - Player keeps Door B -> Tiger: 1/3
- Door C 1/3
  - King opens Door B
    - Player keeps Door A -> Tiger: 1/3

Monty's Dilemma

- Assume gold is behind the first door
- Strategy 2: Player switches doors
- Door A 1/3
  - King opens Door B
    - Player switches Door C -> Tiger: 1/3 * ½
  - King opens Door C
    - Player switches Door B -> Tiger: 1/3 * ½
- Door B 1/3
  - King opens Door C
    - Player switches Door A -> Gold: 1/3
- Door C 1/3
  - King opens Door B
    - Player switches Door A -> Gold: 1/3

Introduction

- Bayes Theorem
- MAP, ML hypothesis
- MAP learners
- Minimum description length principle
- Bayes optimal classifier
- Naïve Bayes learner
- Example: Learning over text data
- Bayesian belief networks
- Expectation maximization algorithm

Two Roles for Bayesian Methods

- Practical learning algorithms
  - Naïve Bayes learning
  - Bayesian belief networks
  - Combine prior knowledge (prior probabilities) with observed data
  - Requires prior probabilities
- Provides useful conceptual framework
  - Provides gold standard for evaluating other learning algorithms
  - Additional insight into Occam's razor

Basic Probability and Bayes Theorem

- Mr. Smith who is correct 75% of the time claims that event X will not occur
- Mr. Jones who is correct 60% of the time claims that event X will occur
- What is p(X)?

Basic Probability and Bayes Theorem

- This problem is underspecified (3 Variables)

Smith	Jones	Actual	Smith Correct	Jones Correct	Probability
N	N	N	Y	Y	P0
N	N	Y	N	N	P1
N	Y	N	Y	N	P2
N	Y	Y	N	Y	P3
Y	N	N	N	Y	P4
Y	N	Y	Y	N	P5
Y	Y	N	N	N	P6
Y	Y	Y	Y	Y	P7

Basic Probability and Bayes Theorem

- p0+p1+p2+...p7 = 1
- Since Smith is correct 75% of the time
  - p0 + p2 + p5 + p7 = 0.75
- Since Jones is correct 60% of the time
  - p0 + p3+p4+p7 = 0.60
- p(X) = p(R=y|S=N and J=Y) =p3/(p2+p3)

Basic Probability and Bayes Theorem

- Let A=p0+p7, B=p1+p6, C=p2+p5, and D=p3+p4, the conditions can

$$A+B+C+D = 1.00$$

$$A \quad +C \quad = 0.75$$

$$A \quad +D = 0.60$$

- Three linear equations with four unknowns.
- Infinitely many solutions.
  - A=0.5, B=0.25, C=0.15, D=0.10
  - A=0.6, B=0.25, C=0.15, D=0.00

Basic Probability and Bayes Theorem

- Even if we arbitrarily select one of these solutions, there are still infinitely many ways of partitioning C and D to give the values of p2 and p3.
- Reasonable assumption to solve this problem?

Basic Probability and Bayes Theorem

- Smith is smarter than Jones. So Smith is correct whenever Jones is correct.
- p3 = 0, p4 = 0. p2 > 0
- So p(X) = 0
- Assume strong correlation between Smith and Jones' answers.

Basic Probability and Bayes Theorem

- Assume that the correctness of the answers are statistically independent.
- Each one of them is likely to be correct independent of whether the other one is correct or not
- 0.6 = (p0+p7)/(p0+p2+p5+p7)=(p3+p4)/(p1+p3+p4+p6)
- 0.75=(p0+p7)/(p0+p3+p4+p7)=(p2+p5)/(p1+p2+p5+p6)

Basic Probability and Bayes Theorem

- Let u = 0.6 and v=0.75, then
  - p0+p7 = uv = 9/20
  - p1+p6 = (1-u)(1-v) = 2/20
  - p2+p5 = (1-u)v = 6/20
  - p3+p4 = u(1-v) = 3/20
- Still not enough to calculate p(X)=p3/(p2+p3)
- Assume symmetry between Y and N
  - p0=p7, p1=p6,p2=p5, and p3=p4
- Prior probability of X is 0.5 and probability of correctness between predicting Y and N is the same

Basic Probability and Bayes Theorem

- p2 = 6/40, p3=3/40 => p(X)=p3/(p2+p3) = 0.33
- Assume that you know that the probability of X a priori is x
  - p0 = uv(1-x) p7 = uvx
  - p1 = (1-u)(1-v)x p6 = (1-u)(1-v)(1-x)
  - p2 = (1-u)v(1-x) p5 = (1-u)vx
  - p3 = u(1-v)x p4 = u(1-v)(1-x)
- Then p(X) = p3/(p2+p3)=u(1-v)x/((1-u)v(1-x)+u(1-v)x)
- Can be extended to more than two predictions
  - p(X)=(pA*pB*pC)/((pA*pB*pC)+(1-pA)(1-pB)(1-pC))

Bayes Theorem

$$P(h|D) = P(D|h)\,P(h) / P(D)$$

P(D) : prior probability of the data D, *evidence*

P(h) : prior probability of the hypothesis h, *prior*

P(h|D) : posterior probability of the hypothesis given the data D, *posterior*

P(D|h) : probability of the data D given the hypothesis h , *likelihood* of the data

Bayes Theorem

$$P(h|D) = P(D|h)\,P(h) / P(D)$$

posterior = likelihood x prior / evidence

By observing the data D we can convert the prior probability P(h) to the a posteriori probability (posterior) P(h|D)

The posterior is probability that h holds after data D has been observed.

The evidence P(D) can be viewed merely as a scale factor that guarantees that the posterior probabilities sum to one.

Choosing Hypotheses

$$P(h|D) = P(D|h)\,P(h) / P(D)$$

Generally want the most probable hypothesis given the training data

Maximum a posteriori hypothesis h_{MAP}

$$h_{MAP} = \operatorname{argmax}_{h \in H} P(h|D)$$
$$= \operatorname{argmax}_{h \in H} P(D|h)\,P(h) / P(D)$$
$$= \operatorname{argmax}_{h \in H} P(D|h)\,P(h)$$

If the priors of hypothesis are equally likely $P(h_i) = P(h_j)$ then one can choose the *maximum likelihood* (ML) hypothesis

$$h_{ML} = \operatorname{argmax}_{h \in H} P(D|h)$$

Example: Bayes Theorem

A patient takes a lab test and the result is positive. The test returns a correct positive ($\oplus$) result in 98% of the cases in which the disease is actually present, and a correct negative ($\varnothing$) result in 97% of the cases in which the disease is not present. Furthermore, 0.8% of the entire population have the disease. Hypotheses : cancer, ¬cancer

priors $P(h)$: $P(\text{cancer}) = 0.008$, $P(\neg \text{cancer}) = 0.992$

likelihoods $P(D|h)$: $P(\oplus|\text{cancer}) = 0.98$, $P(\varnothing|\text{cancer}) = 0.02$
$P(\oplus|\neg\text{cancer}) = 0.03$, $P(\varnothing|\neg\text{cancer}) = 0.97$

Maximum posteriors argmax $P(h|D)$:

$P(\text{cancer}|\oplus) \sim P(\oplus|\text{cancer})P(\text{cancer}) = 0.0078$

$P(\neg\text{cancer}|\oplus) \sim P(\oplus|\neg\text{cancer})\,P(\neg\text{cancer}) = 0.0298$

$P(\text{cancer}|\oplus) = 0.0078/(0.0078+0.0298) = 0.21$

$P(\neg\text{cancer}|\oplus) = 0.0298/(0.0078+0.0298) = 0.79$

Bayes Formulas for Probabilities

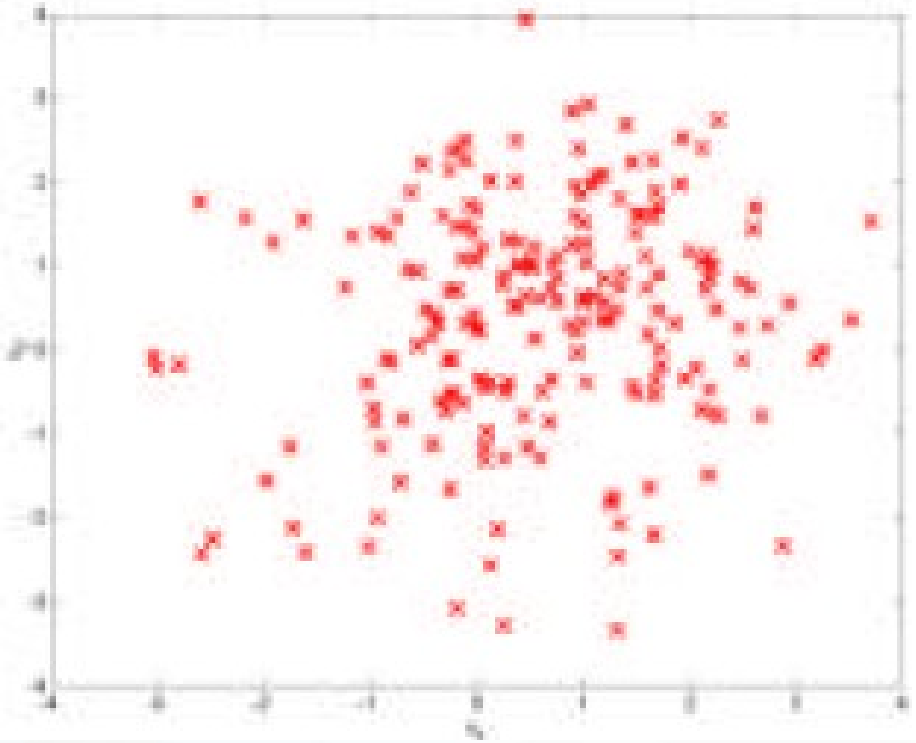
- Product rule:
  - $p(A * B) = p(A|B)*p(B) = p(B|A)*p(A)$
- Sum rule:
  - $p(A+B) = p(A) + p(B) – p(A*B)$
- Bayes Theorem:
  - $p(h|D) = p(D|h)P(h)/p(D)$
- Theorem of total probability: if events Ai to An are mutually exclusive with sum p(Ai) = 1
  - $p(B) = \text{sum } p(B|Ai)*p(A)$

Bayes Theorem Example

$$P(x_1, x_2 | \mu_1, \mu_2, \sigma) = 1/(2\pi\sigma) \exp -\Sigma_i (x_i - \mu_i)^2 / 2\sigma^2$$

$$h = \{\mu_1, \mu_2, \sigma\}$$

$$D = \{x_1, \ldots, x_m\}$$



Gaussian Probability Function

- $P(D|\mu_1,\mu_2,\sigma) = \Pi_m P(x^m|\mu_1,\mu_2,\sigma)$
- Maximum likelihood hypothesis h_{ML}

  $h_{ML} = \text{argmax}_{\mu 1,\mu 2,\sigma} P(D|\mu_1,\mu_2,\sigma)$
- Trick: maximize log-likelihood

$\log P(D|\mu_1,\mu_2,\sigma) = \Sigma_m \log P(x^m|\mu_1,\mu_2,\sigma)$

$= \Sigma_m \log (1/(2\pi\sigma) \exp -\Sigma_i (x^m_i-\mu_i)^2/2\sigma^2$

$= -M \log (2\pi\sigma) - \Sigma_m \Sigma_i (x^m_i-\mu_i)^2/2\sigma^2$

Gaussian Probability Function

$$\partial \log P(D|\mu_1,\mu_2,\sigma)/\partial \mu_i = 0$$

$$\Sigma_m x^m_i-\mu_i = 0 \Rightarrow \mu_{i\,ML} = 1/M \Sigma_m x^m_i = E[x^m]$$

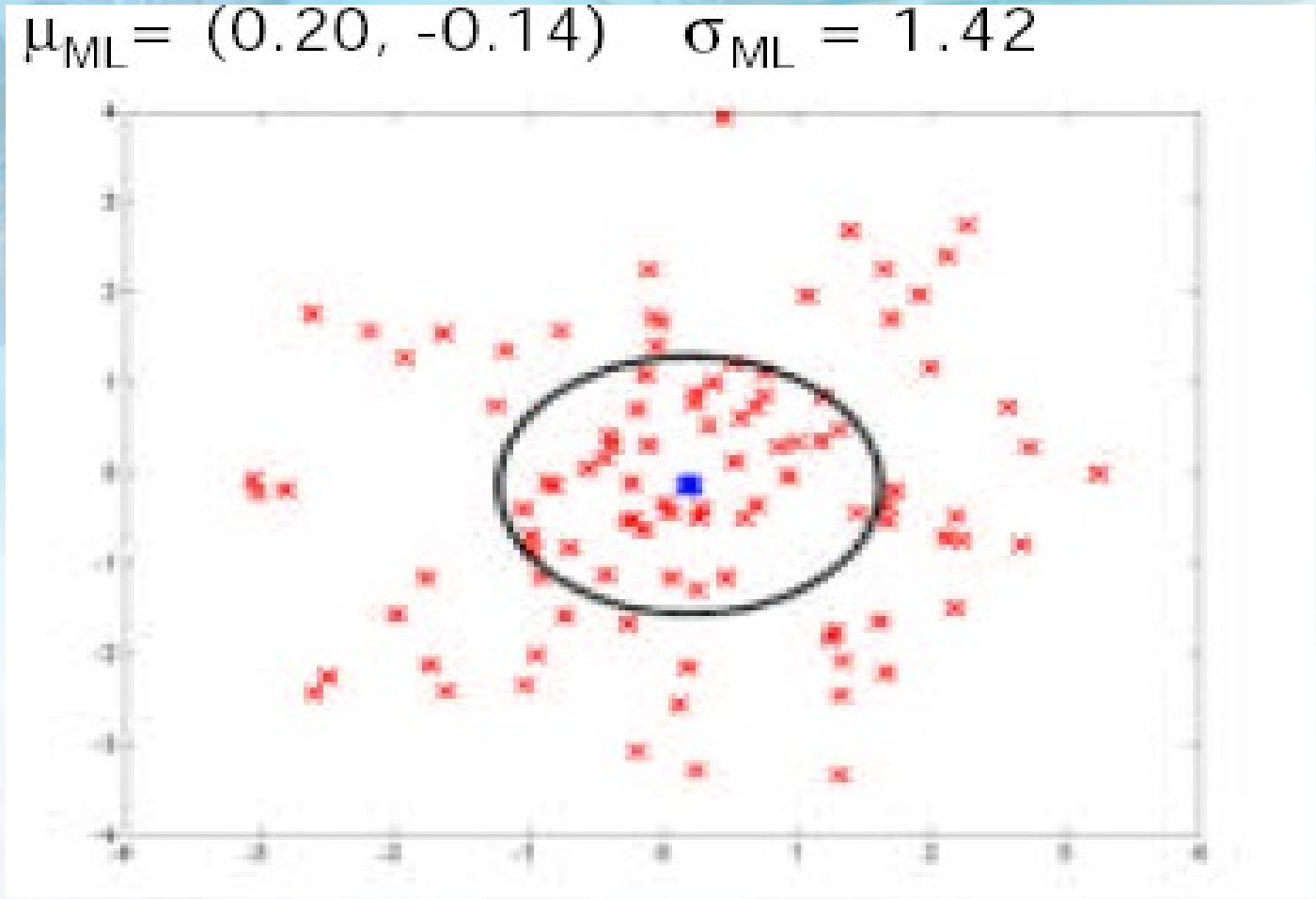
$$\partial \log P(D|\mu_1,\mu_2,\sigma)/\partial \sigma = 0$$

$$\sigma_{ML} = \Sigma_m \Sigma_i (x^m_i-\mu_i)^2 / 2M = E[(\Sigma_i (x^m_i-\mu_i)^2] / 2$$

Maximum likelihood hypothesis $h_{ML} = \{\mu_{iML},\sigma_{ML}\}$

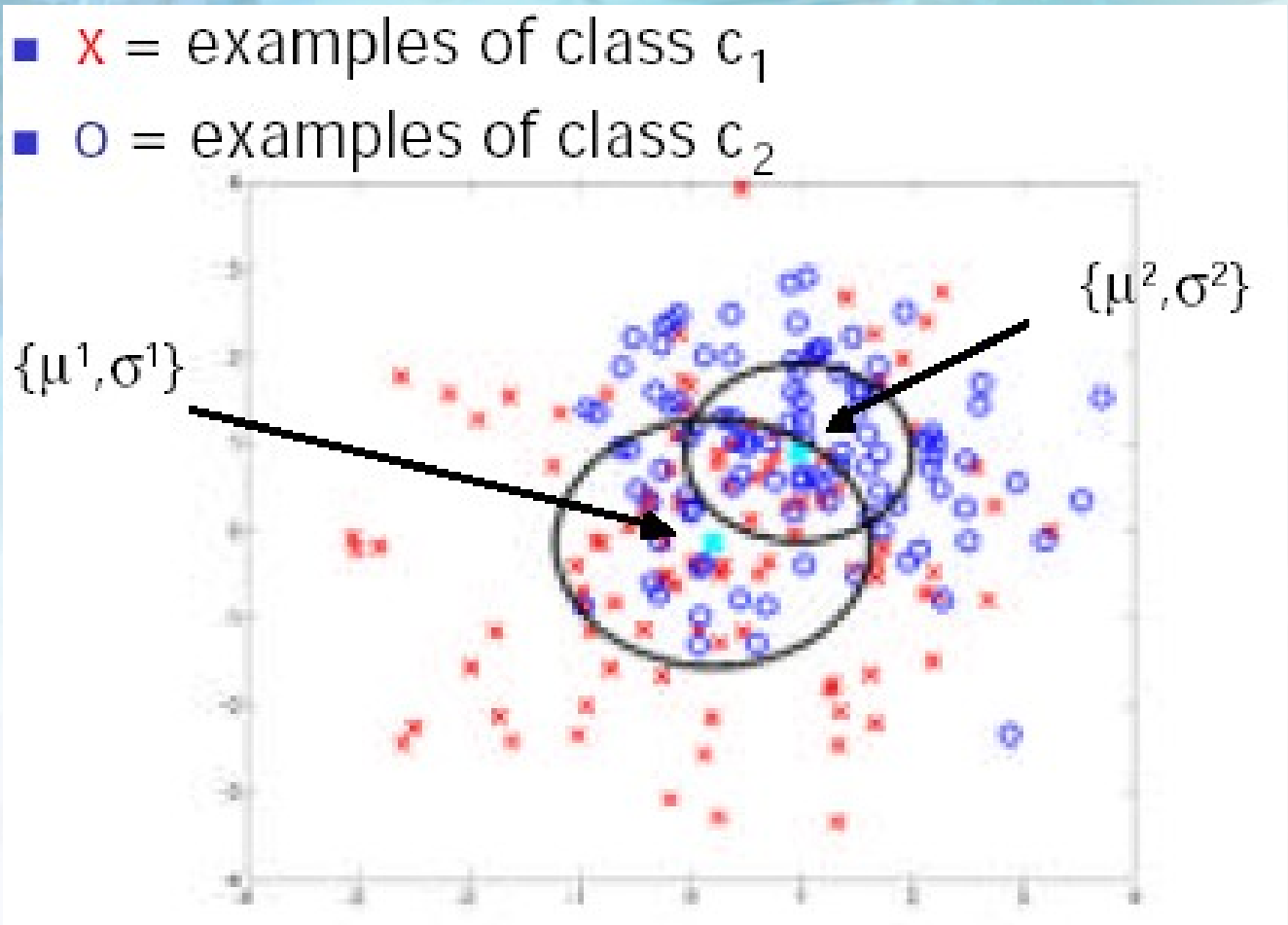
Maximum Likelihood Hypothesis

$\mu_{ML} = (0.20, -0.14)$ $\sigma_{ML} = 1.42$



Bayes Decision Rule

- x = examples of class c_1
- o = examples of class c_2



Bayes Decision Rule

Assume we have two Gaussians distributions associated to two separate classes c_1, c_2.

$P(x|c_i) = P(x|\mu^i,\sigma^i) = 1/(2\pi\sigma)\ \exp -\Sigma_i\ (x_i-\mu_i)^2/2\sigma^2$

Bayes decision rule (max posterior probability)

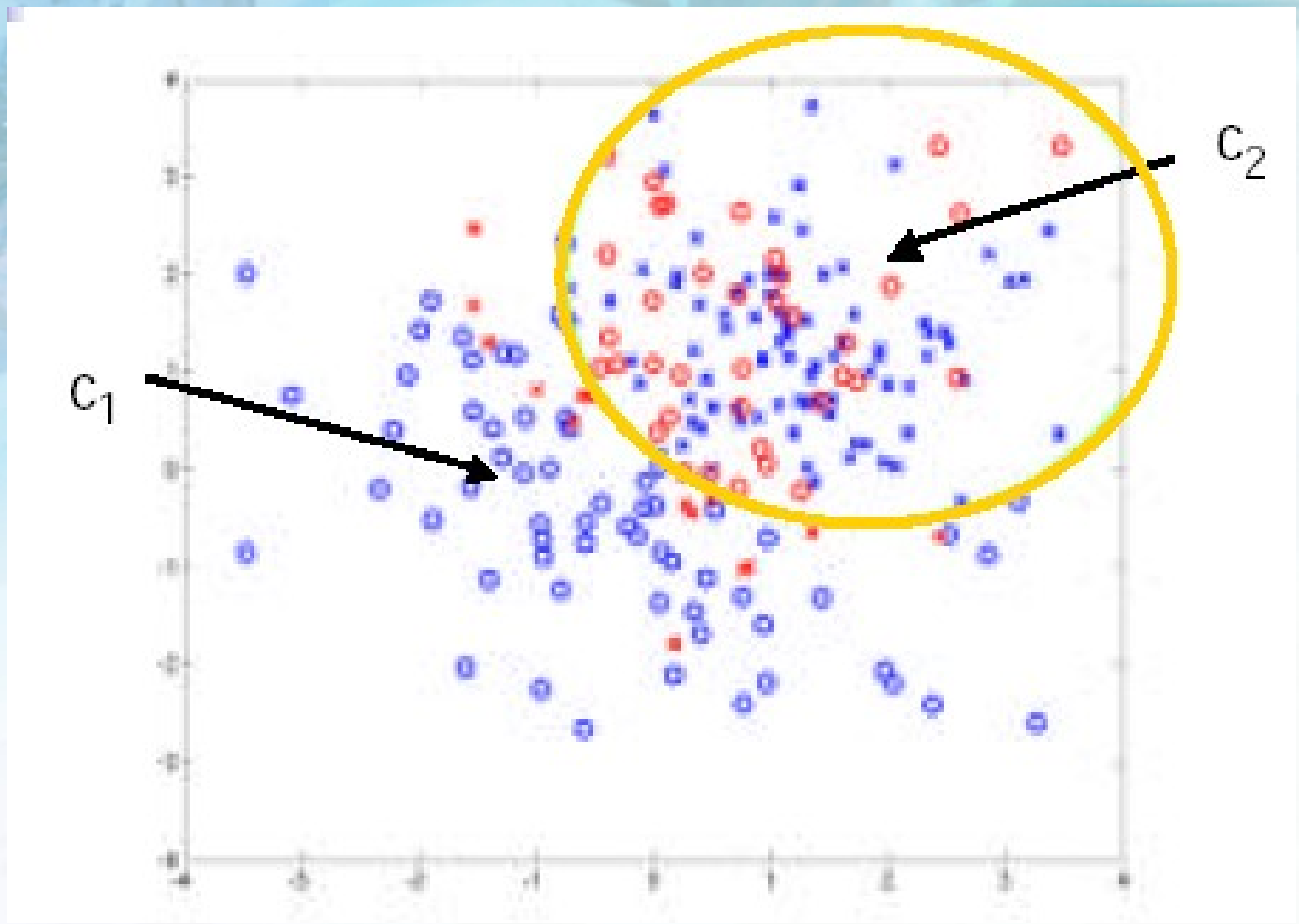
Decide c_1 if $P(c_1|x) > P(c_2|x)$

otherwise decide c_2.

if $P(c_1) = P(c_2)$ use maximum likelihood $P(x|c_i)$

else use maximum posterior $P(c_i|x) = P(x|c_i)\ P(c_i)$

Bayes Decision Rule



Two Category Case

- Discriminant functions:

  if $g(x) > 0$ then c_1 else c_2

- $g(x) = P(c_1|x) - P(c_2|x)$

  $= P(x|c_1)\,P(c_1) - P(x|c_1)\,P(c_1)$

- $g(x) = \log P(c_1|x) - \log P(c_2|x)$

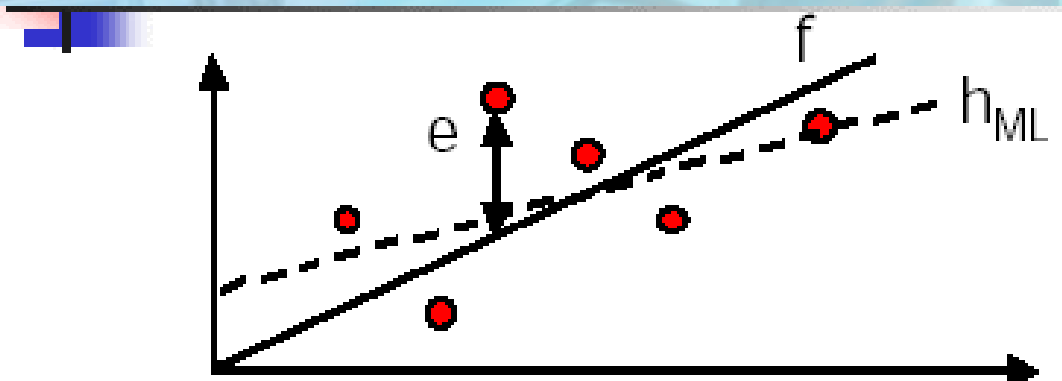
  $= \log P(x|c_1)/P(x|c_2) - \log P(c_1)/P(c_2)$

- Gaussian probability functions with identical σ_i

$g(x) = (x-\mu_2)^2/2\sigma^2 - (x-\mu_1)^2/2\sigma^2 + \log P(c_1) - \log P(c_2)$

decision surface is a line/hyperplane

Learning a Real Valued Function



- Consider a real-valued target function f
- Noisy training examples $<x_i, d_i>$

$$d_i = f(x_i) + e_i$$

e_i is a random variable drawn from a Gaussian distribution with zero mean.

- The maximum likelihood hypothesis h_{ML} is the one that minimizes the squared sum of errors

$$h_{ML} = \text{argmin}_{h \in H} \sum_i (d_i - h(x_i))^2$$

Learning a Real Valued Function

$$
\begin{aligned}
h_{ML} &= \underset{h \in H}{\operatorname{argmax}}\, p(D|h) \\
&= \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} p(d_i|h) \\
&= \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2}
\end{aligned}
$$

Maximize natural log of this instead...

$$
\begin{aligned}
h_{ML} &= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \\
&= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} -\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \\
&= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} -\left(d_i - h(x_i)\right)^2 \\
&= \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} \left(d_i - h(x_i)\right)^2
\end{aligned}
$$

Brute Force MAP Hypothesis Learner

- For each hypothesis in H compute the posterior probability

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- Output hypothesis h with the maximum posterior probability

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}} P(h|D)$$

Relation to Concept Learning

- Consider usual concept learning task
  - Instance space X, hypothesis space H, and training examples D
  - Consider the FIND-S learning algorithm (Outputs most specific hypothesis in the version space $VS_{H,D}$)
- What would Bayes rule output as the MAP hypothesis?
- Does FIND-S output the MAP hypothesis?

Relation to Concept Learning

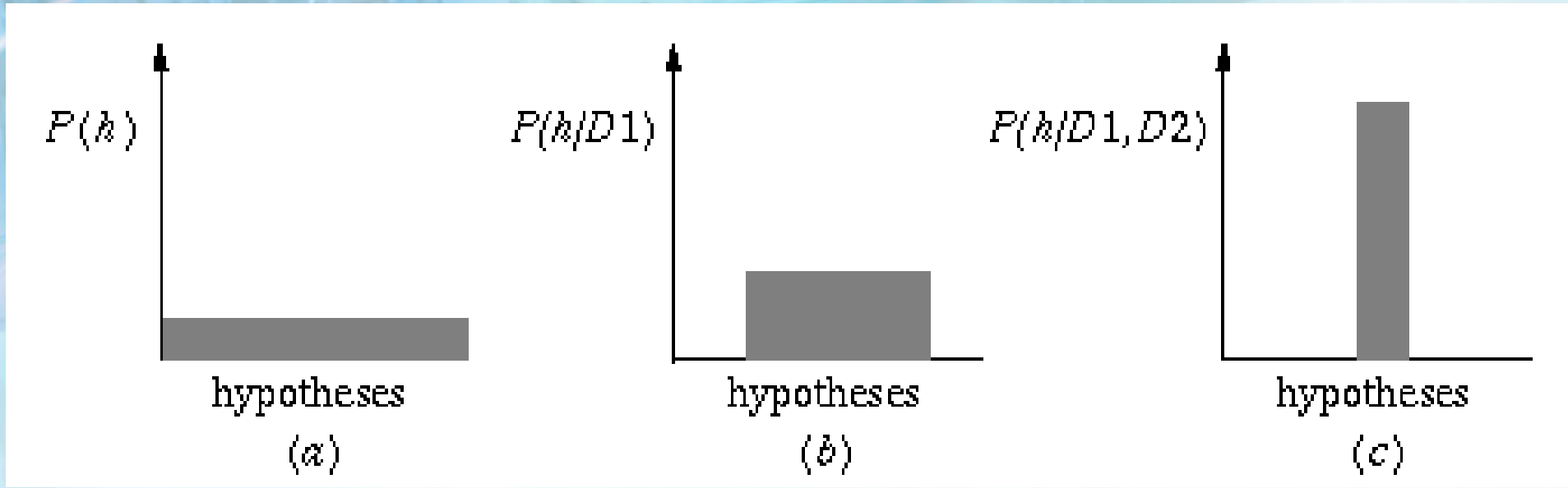
- Assume fixed set of instances ($x_1 \ldots x_m$)
- Assume D is the set of classifications
  - $D=<c(x_1),\ldots,c(x_m)>$
- Choose P(D|h)

Relation to Concept Learning

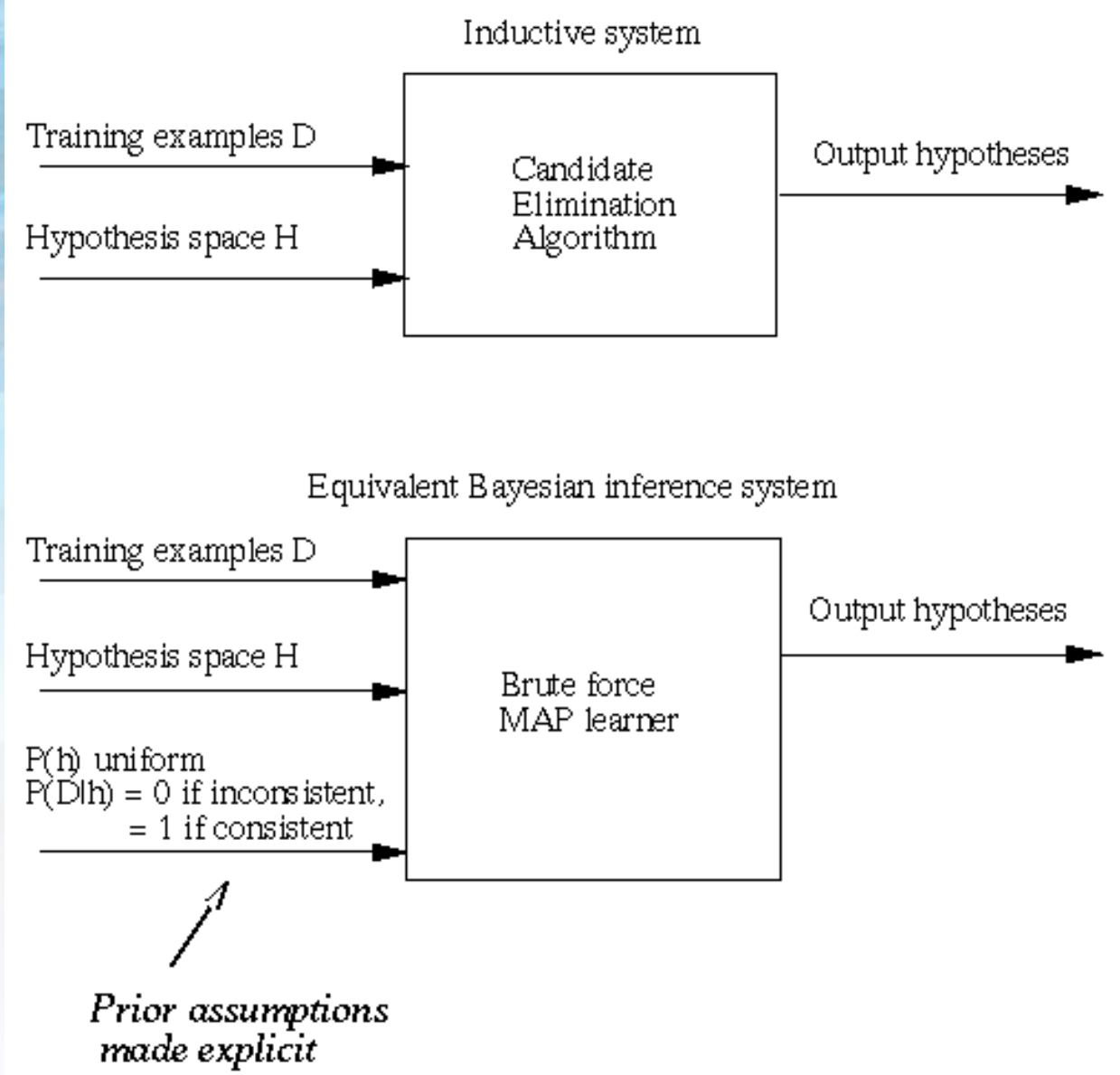
- Choose P(D|h)
  - P(D|h) = 1 if h is consistent with D
  - P(D|h) = 0 otherwise
- Choose P(h) be the uniform distribution
  - P(h) = 1/||H|| for all h in H
- Then

$$P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$

Evolution of Posterior Probabilities



Characterizing Learning Algorithms by Equivalent MAP Learners



Learning to Predict Probabilities

- Consider predicting survival probabilities from patient data
- Training examples $\langle x_i, d_i \rangle$ where $d_i = 0$ or 1
- Want to train neural net to output a probability given x_i (not 0 or 1)
- In this case, we can show

Learning to Predict Probabilities

- Maximum likelihood hypothesis:

$$h_{ML} = \text{argmax}_{h \in H} \sum_i d_i \ln h(x_i) + (1-d_i) \ln (1-h(x_i))$$

maximize *cross entropy* between d_i and $h(x_i)$

- Weight update rule for synapses w_k to output neuron $h(x_i)$

$$w_k = w_k + \eta \sum_i (d_i - h(x_i))\, x_k$$

- Compare to standard BP weight update rule

$$w_k = w_k + \eta \sum_i h(x_i)(1-h(x_i))\,(d_i - h(x_i))\, x_k$$

Minimum Description Length Principle

- Occam's razor: Prefer simplest (shortest) hypothesis
- MDL: prefer the hypothesis that minimizes $h_{MDL} = \text{argmin}_{h \text{ in } H} L_{C1}(h) + L_{C2}(D|h)$ where $L_C(x)$ is the description length of x under encoding C

Minimum Description Length Principle

Example: H = decision trees, D = training data labels

- $L_{C_1}(h)$ is # bits to describe tree h
- $L_{C_2}(D|h)$ is # bits to describe D given h
  - Note $L_{C_2}(D|h) = 0$ if examples classified perfectly by h. Need only describe exceptions
- Hence h_{MDL} trades off tree size for training errors

Minimum Description Length Principle

$$
\begin{aligned}
h_{MAP} &= \arg\max_{h \in H} P(D|h)P(h) \\
&= \arg\max_{h \in H} \log_2 P(D|h) + \log_2 P(h) \\
&= \arg\min_{h \in H} -\log_2 P(D|h) - \log_2 P(h) \quad (1)
\end{aligned}
$$

Interesting fact from information theory:

> The optimal (shortest expected coding length) code for an event with probability p is $-\log_2 p$ bits.

So interpret (1):

- $-\log_2 P(h)$ is length of h under optimal code
- $-\log_2 P(D|h)$ is length of D given h under optimal code

$\rightarrow$ prefer the hypothesis that minimizes

$$length(h) + length(misclassifications)$$

Most Probable Classification of New Instances

- So far we sought the most probable *hypothesis* h_{MAP}?
- What is most probable *classification* of a new instance x given the data D?

$h_{MAP}(x)$ is not the most probable classification, although often a sufficiently good approximation of it.

- Consider three possible hypotheses:
- $P(h_1|D) = 0.4$, $P(h_2|D) = 0.3$, $P(h_3|D) = 0.3$
- Given a new instance x, $h_1(x)=+$, $h_2(x)=-$, $h_3(x)=-$

$$h_{MAP}(x) = h_1(x) = +$$

- most probable classification:

$P(+)=P(h_1|D)=0.4$ $\quad$ $P(-)=P(h_2|D) + P(h_3|D) = 0.6$

Bayes Optimal Classifier

- $c_{max} = \text{argmax}_{c_j \in C} \sum_{h_i \in H} P(c_j|h_i)\ P(h_i|D)$
- Example:

  $P(h_1|D) = 0.4,\ P(h_2|D) = 0.3,\ P(h_3|D) = 0.3$

  $P(+|h_1)=1,\ P(-|h_1)=0$

  $P(+|h_2)=0,\ P(-|h_2)=1$

  $P(+|h_3)=0,\ P(-|h_3)=1$

  therefore

  $\sum_{h_i \in H} P(+|h_i)\ P(h_i|D) = 0.4$

  $\sum_{h_i \in H} P(-|h_i)\ P(h_i|D) = 0.6$

  $\text{argmax}_{c_i \in C} \sum_{h_i \in H} P(v_i|h_i)\ P(h_i|D) = -$

Gibbs Classifier

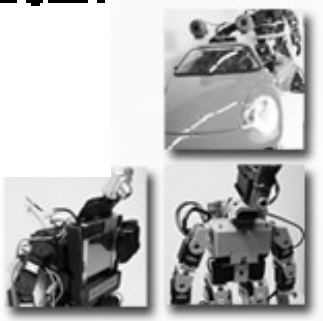
Bayes optimal classifier provides best result, but can be expensive if many hypotheses.

Gibbs algorithm:

1. Choose one hypothesis at random, according to $P(h|D)$
2. Use this to classify new instance

Surprising fact: Assume target concepts are drawn at random from H according to priors on H. Then:

$$E[error_{Gibbs}] \leq 2E[error_{BayesOptimal}]$$



Gibbs Classifier

Suppose correct, uniform prior distribution over H, then

- Pick any hypothesis from VS, with uniform probability
- Its expected error no worse than twice Bayes optimal

Bayes vs. MAP Method

- The maximum posterior hypothesis estimates an *point* h_{MAP} in the hypothesis space H.
- Bayes method instead estimates and uses a complete *distribution* P(h|D).
- The difference appears when inference MAP or Bayes method are used for inference of unseen instances and one compares the distributions P(x|D)
- MAP: $P(x|D) = h_{MAP}(x)$ with $h_{ML} = \text{argmax}_{h \in H} P(h|D)$
- Bayes: $P(x|D) = \Sigma_{h_i \in H} P(x|h_i) P(h_i|D)$
- For reasonable prior distributions P(h) MAP and Bayes solution are equivalent in the asymptotic limit of infinite training data D.

Naïve Bayes Classifier

- Very practical learning method (along with decision trees, nearest neighbor, neural networks)
- When to use
  - Moderate or large training data
  - Attributes that describe instances x are conditionally independent given classification
- Successful applications
  - Medical diagnosis
  - Classifying text documents
  - Keyphrase extraction
  - Spam email

Naïve Bayes Classifier

- Assume discrete target function F: $X \rightarrow C$, where each instance x described by attributes $<a_1, a_2, \ldots, a_n>$
- Most probable value of f(x) is:

$$c_{MAP} = \text{argmax}_{c_j \in C} P(c_j | <a_1, a_2, \ldots, a_n>)$$
$$= \text{argmax}_{c_j \in C} P(<a_1, a_2, \ldots, a_n> | c_j) P(c_j) / P(<a_1, a_2, \ldots, a_n>)$$
$$= \text{argmax}_{c_j \in C} P(<a_1, a_2, \ldots, a_n> | c_j) P(c_j)$$

- Naïve Bayes assumption: $P(<a_1, a_2, \ldots, a_n> | c_j) = \prod_i P(a_i | c_j)$

$$c_{NB} = \text{argmax}_{c_j \in C} P(c_j) \prod_i P(a_i | c_j)$$

Naïve Bayes Algorithm

Naïve_Bayes_Learn(examples)

for each target valuecj estimate $P(c_j)$

for each attribute value ai estimate of each attribute a estimate $P(a_i|c_j)$

Classify_New_Instance(x)

$$c_{NB} = \text{argmax}_{c_j \in C} P(c_j) \prod_{a_i \in X} P(a_i|c_j)$$

Naïve Bayes Algorithm Example

- Consider *PlayTennis* and new instance

<Outlook=Sunny, Temp=cool, Humidity=high, Wind=strong>

- Compute $c_{NB} = \text{argmax}_{c_j \in C} P(c_j) \prod_{a_i \in x} P(a_i|c_j)$

playtennis (9+,5-)

P(yes) = 9/14, P(no) = 5/14

wind=strong (3+,3-)

P(strong|yes) = 3/9 , P(strong|no) 3/5

...

P(yes) P(sun|yes) P(cool|yes) P(high|yes) P(strong|yes)= 0.005

P(no) P(sun|no) P(cool|no) P(high|no) P(strong|no)= 0.021

Naïve Bayes: Subtleties

Conditional independence assumption is often violated

$$P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$$

- ...but it works surprisingly well anyway. Note don't need estimated posteriors $\hat{P}(v_j|x)$ to be correct; need only that

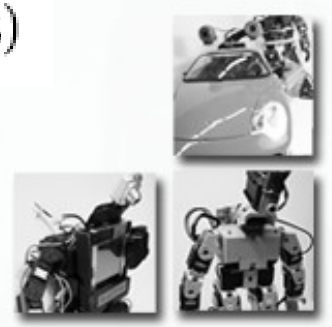
$$\underset{v_j \in V}{\operatorname{argmax}} \hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) P(a_1 \ldots, a_n | v_j)$$

- see [Domingos & Pazzani, 1996] for analysis
- Naive Bayes posteriors often unrealistically close to 1 or 0

Naïve Bayes: Laplacian Smoothing

- What if none (n_c=0) of the training instances with target value c_j have attribute a_i?

  $P(a_i|c_j) = n_c/n = 0$ and $P(c_j) \prod_{a_i \in x} P(a_i|c_j) = 0$
- Solution: Bayesian estimate for $P(a_i|c_j)$
- $P(a_i|c_j) = (n_c + mp)/(n + m)$
  - n : number of training examples for which c=cj
  - n_c : number of examples for which c=cj and a=ai
  - p : prior estimate of $P(a_i|c_j)$
  - m : weight given to prior (number of "virtual" examples)



Learning to Classify Text

- Learn which news articles are of interest
- Learn to classify spam
- Learn to classify web pages by topic
- Naïve Bayes is amongst the most effective algorithms
- What attributes should we use to classify text?

Learning to Classify Text

Target concept $Interesting? : Document \rightarrow \{+, -\}$

1. Represent each document by vector of words
   - one attribute per word position in document
2. Learning: Use training examples to estimate
   - $P(+)$
   - $P(-)$
   - $P(doc|+)$
   - $P(doc|-)$

Naive Bayes conditional independence assumption

$$P(doc|v_j) = \prod_{i=1}^{length(doc)} P(a_i = w_k|v_j)$$

where $P(a_i = w_k|v_j)$ is probability that word in position i is w_k, given v_j

one more assumption:

$$P(a_i = w_k|v_j) = P(a_m = w_k|v_j), \forall i, m$$

Learning to Classify Text

LEARN_NAIVE_BAYES_TEXT($Examples, V$)

1. *collect all words and other tokens that occur in Examples*

- $Vocabulary \leftarrow$ all distinct words and other tokens in $Examples$

2. *calculate the required $P(v_j)$ and $P(w_k|v_j)$ probability terms*

- For each target value v_j in V do
  - $docs_j \leftarrow$ subset of $Examples$ for which the target value is v_j
  - $P(v_j) \leftarrow \frac{|docs_j|}{|Examples|}$
  - $Text_j \leftarrow$ a single document created by concatenating all members of $docs_j$
  - $n \leftarrow$ total number of words in $Text_j$ (counting duplicate words multiple times)
  - for each word w_k in $Vocabulary$
    - $n_k \leftarrow$ number of times word w_k occurs in $Text_j$
    - $P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$

Learning to Classify Text

CLASSIFY_NAIVE_BAYES_TEXT(Doc)

- $positions \leftarrow$ all word positions in Doc that contain tokens found in $Vocabulary$
- Return v_{NB}, where

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_{i \in positions} P(a_i | v_j)$$

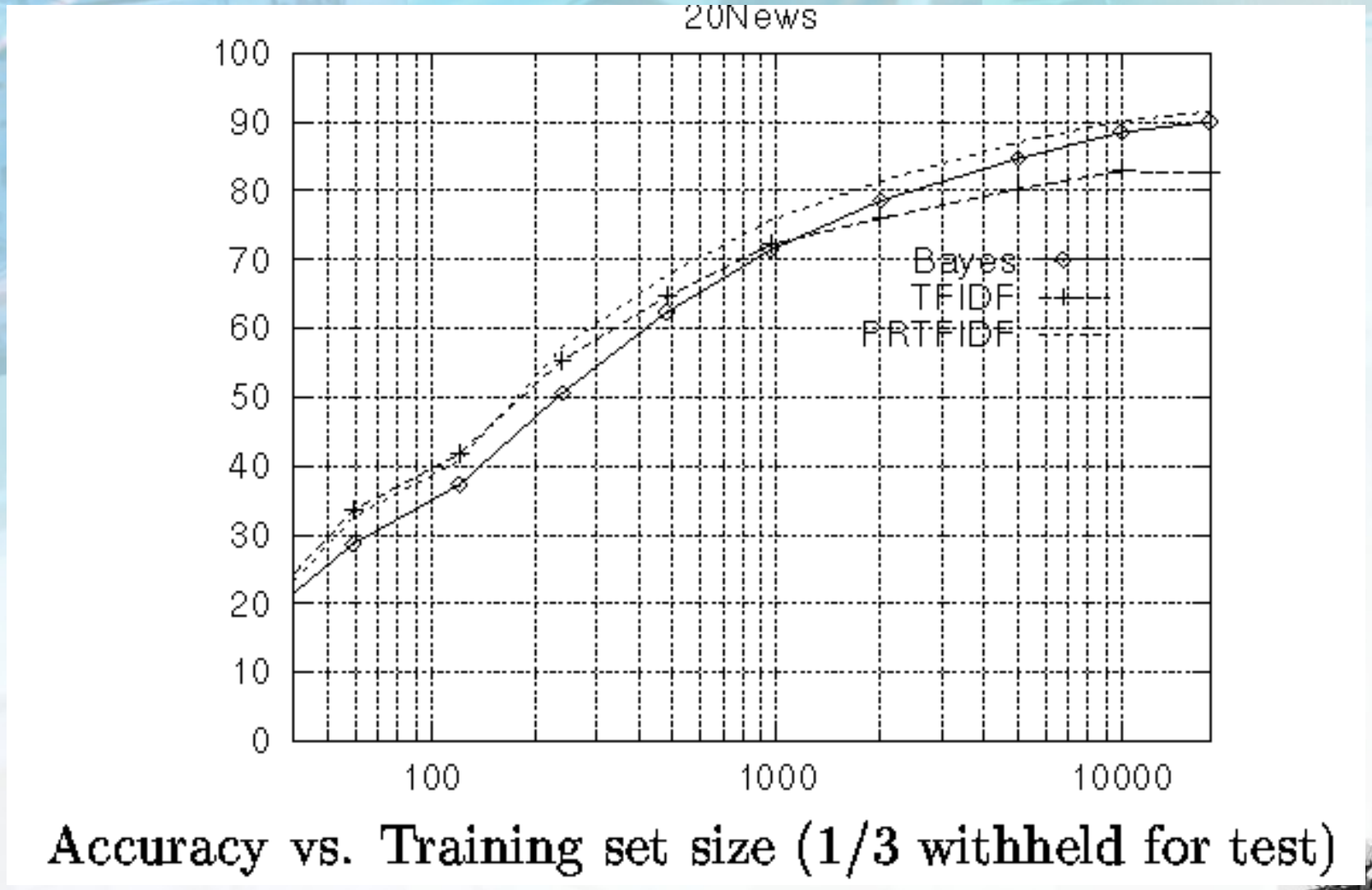
Learning to Classify Text: Preprocessing

- Eliminate stopwords such as the, an, with, nevertheless, …
  - Create list of most commonly used English words
- Stemming
  - Convert a word into its root word
  - Feels → feel
  - Feeling → feel
  - Felt → feel

Twenty Newsgroups

- Given 1000 training examples from each newsgroup
- Learn to classify new documents according to which newsgroup it came from
- Naïve Bayes: 89% classification accuracy

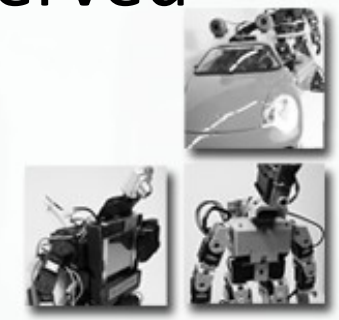
Learning Curve for 20 Newsgroups



Accuracy vs. Training set size (1/3 withheld for test)

Bayesian Belief Networks

- Interesting because
  - Naïve Bayes assumption of conditional independence is too restrictive
  - Intractable without such assumptions
  - Bayesian belief networks describe conditional dependence between a subset of variables
- Allows combining prior knowledge about (in)dependencies among variables with observed training data
- Also called Bayes Nets



Conditional Independence

Definition: X is *conditionally independent* of Y given Z if the probability distribution governing X is independent of the value of Y given the value of Z; that is, if

$$(\forall x_i, y_j, z_k)\ P(X = x_i | Y = y_j, Z = z_k) = P(X = x_i | Z = z_k$$

more compactly, we write

$$P(X|Y, Z) = P(X|Z)$$

Conditional Independence

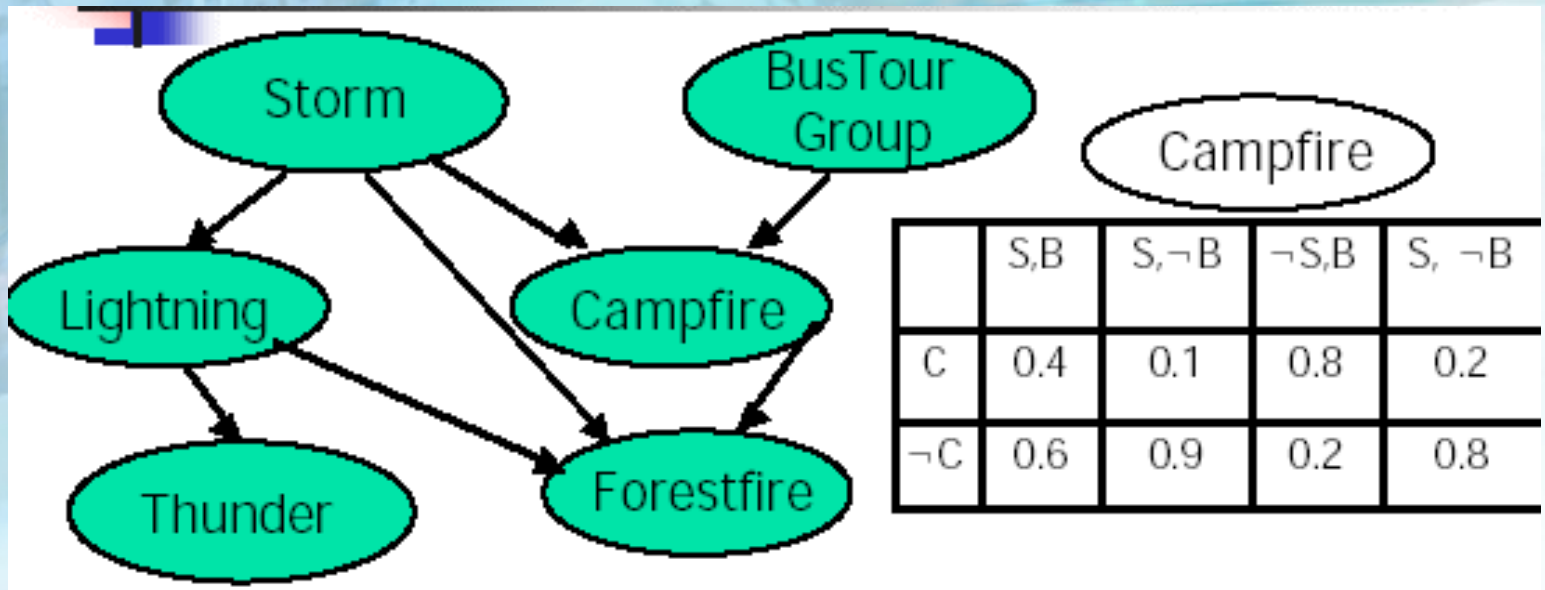
Example: *Thunder* is conditionally independent of *Rain*, given *Lightning*

$$P(Thunder|Rain, Lightning) = P(Thunder|Lightning)$$

Naive Bayes uses cond. indep. to justify

$$\begin{aligned} P(X, Y|Z) &= P(X|Y, Z)P(Y|Z) \\ &= P(X|Z)P(Y|Z) \end{aligned}$$

Bayesian Belief Network

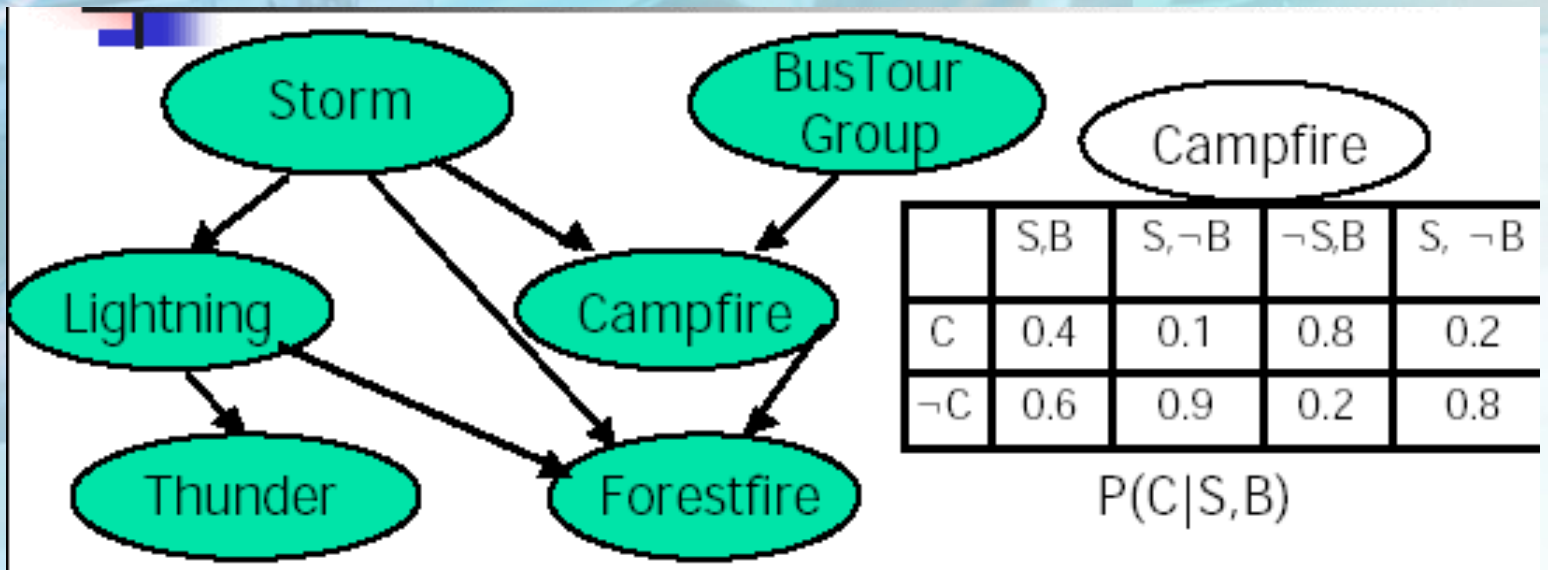


	S,B	S,¬B	¬S,B	¬S,¬B
C	0.4	0.1	0.8	0.2
¬C	0.6	0.9	0.2	0.8

Network represents a set of conditional independence assertions:

- Each node is conditionally independent of its non-descendants, given its immediate predecessors. (directed acyclic graph)

Bayesian Belief Networks



	S,B	S,¬B	¬S,B	¬S,¬B
C	0.4	0.1	0.8	0.2
¬C	0.6	0.9	0.2	0.8

P(C|S,B)

Network represents joint probability distribution over all variables

- P(Storm,BusGroup,Lightning,Campfire,Thunder,Forestfire)
- $P(y_1,\ldots,y_n) = \prod_{i-1}^{n} P(y_i|\text{Parents}(Y_i))$
- joint distribution is fully defined by graph plus $P(y_i|\text{Parents}(Y_i))$

Bayesian Belief Networks Example

- Mary walks outside and finds that the street and lawn are wet. She concludes that is has rained recently. Furthermore, she decides that she does not need to water the roses

Rain or sprinkler --> street = wet

Rain or sprinkler --> lawn = wet

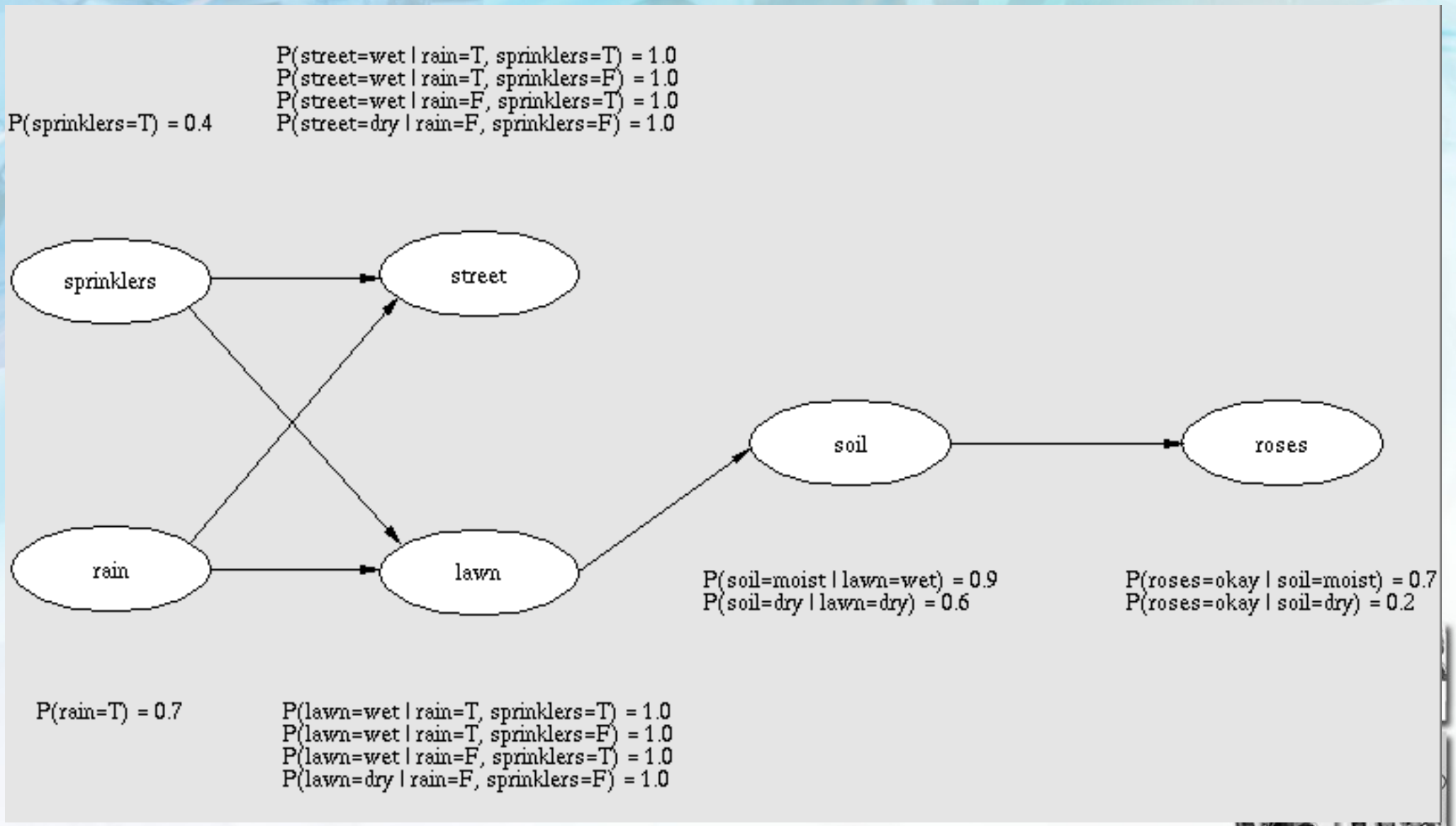
Lawn = wet --> soil = moist

Soil = moist --> roses = okay

Bayesian Belief Network Example

- Convert into a graph showing the dependencies
- Each variable is considered as a random variable with states true and false
- Conditional probability tables to represent the knowledge of the world

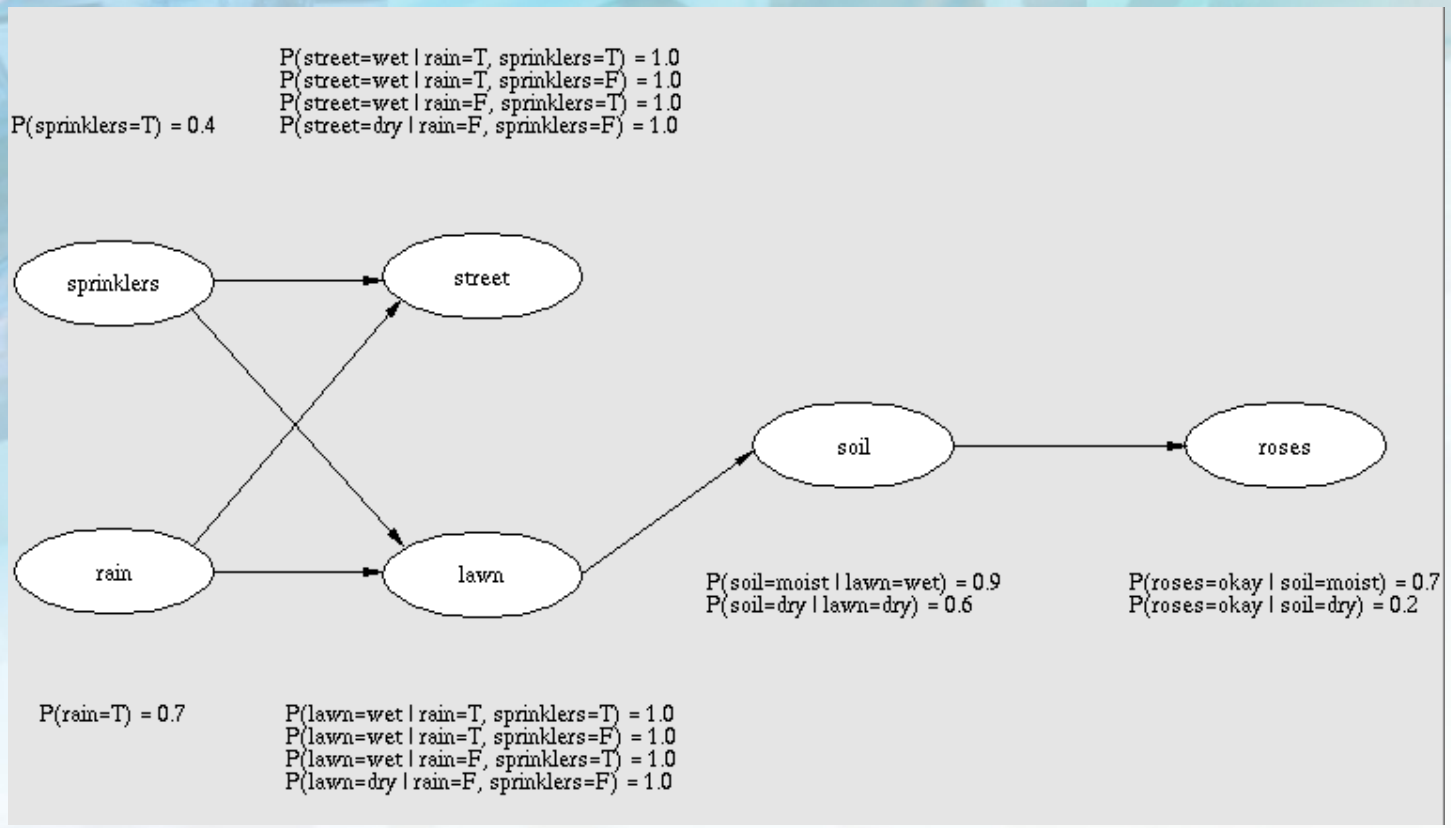
Bayesian Belief Network Example



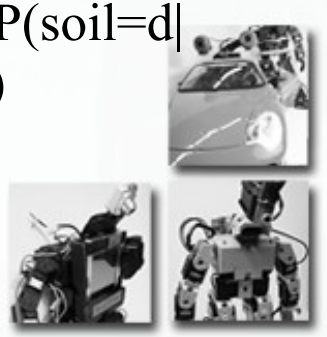
Bayesian Belief Networks Example

- What is the probability of a world where
  - Roses = okay
  - Soil = dry
  - Lawn = wet
  - Street = wet
  - Sprinklers = no
  - Raining = yes

Bayesian Belief Network Example



P(spr=f, rain=t, street=w, lawn=w, soil=d, roses=ok) = P(roses=ok|soil=d)*P(soil=d|lawn=w)*P(lawn=w|rain=t,spr=f)*P(str=w|rain=t,spr=f)*P(spr=f)*P(rain=t)
= 0.0084



Bayesian Belief Networks Example

- Belief Updating: compute values for random variables
  - Example: What is the probability of the lawn being wet given that the roses are okay
  - P(lawn=dry|roses=ok) = 0.1190
  - P(lawn=wet|roses=ok) = 0.8810
- Belief Revision: find global assignment which maximizes probability
  - What is the best assignment for all variables given that the roses are okay
  - P(spr=f, rain=t, street=w,lawn=wet,soil=w,roses=okay) = 0.26

Inference in Bayesian Nets

- Infer (probabilities) values of one or more variables given observed values of others
  - Bayes net contains all information needed for this inference
  - If single variable with unknown value, easy to infer it
  - In general, the problem is NP hard
- In practice, can succeed in many cases
  - Exact inference methods work well for some network structures
  - Monte Carlo methods simulate the network randomly to calculate approximate solutions

Learning of Bayesian Belief Networks

- Several aspects of learning task
  - Network structure may be known or unknown
  - Training examples might provide values of all or some network variables
- If structure known and observe all variables
  - Easy as training a naïve Bayes classifier

Learning of Bayes Nets

- Suppose structure known, variables partially observable
  - e.g., observe ForestFire, Storm, BusTourGroup, Thunder, but not Lightning, CampFire, ...
  - Similar to training neural nets with hidden units
  - In fact, can learn network conditional probability tables using gradient ascent
- Converge to network h that locally maximizes P(D|h)

Gradient Ascent for Bayes Nets

- Let w_{ijk} denote one entry in the conditional probability table for variable Y_i in the network

$$w_{ijk} = P(Y_i = y_{ij} | \text{Parents}(Y_i) = \text{list } u_{ik} \text{ of values})$$

e.g. : if Y_i=Campfire, then u_{ik} might be

<Storm=True, BusTourGroup=False>

Perform gradient ascent on ln P(D|h) by repeatedly

1. Update all w_{ijk} using training data D

$$w_{ijk} = w_{ijk} + \eta \sum_{d \in D} P_h(y_{ij}, u_{ik} | d) / w_{ijk}$$

2. Renormalize the wijk to assure

$$\Sigma_j w_{ijk} = 1$$

$$w_{ijk} \leq 1$$

Learning of Bayes Nets

- EM algorithm can also be used. Repeatedly:
  - Calculate probabilities of unobserved variables assuming h
  - Calculate new wijk to maximize E[ln P(D|h)] where D now includes observed and calculated probabilities of unobserved variables
- When structure unknown
  - Algorithms use greedy search to add/subtract edges and nodes
  - Active research topic

Summary: Bayesian Belief Networks

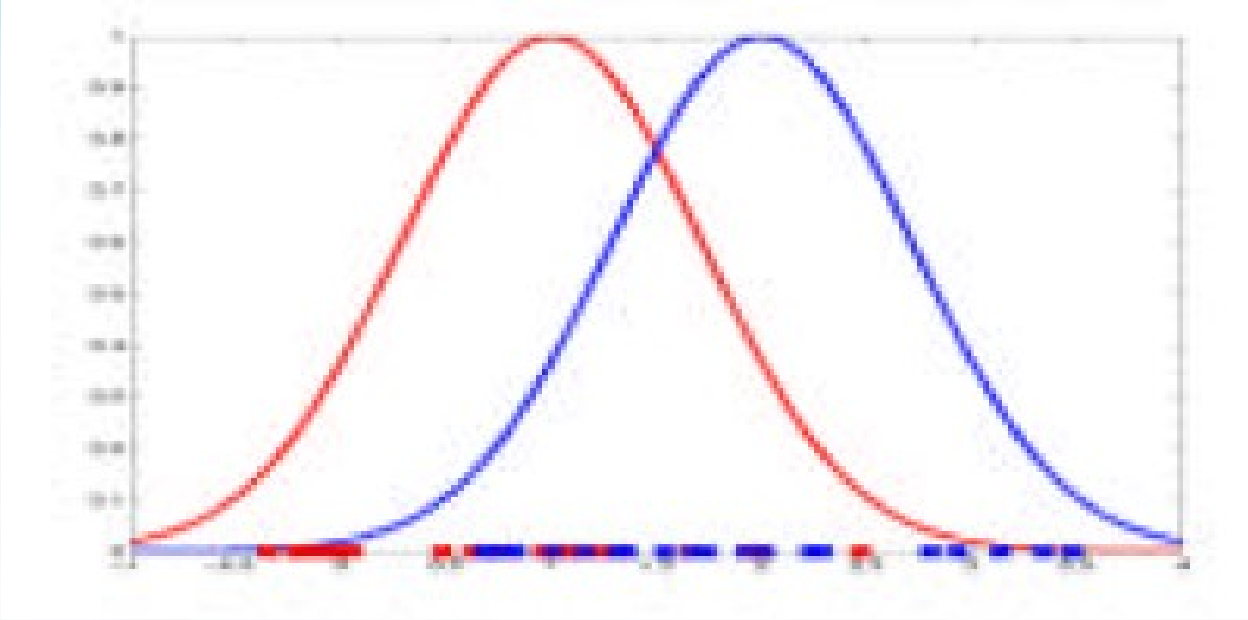
- Combine prior knowledge with observed data
- Impact of prior knowledge (when correct?) is to lower the sample complexity
- Active research areas
  - Extend from boolean to real-valued variables
  - Parameterized distributions instead of tables
  - Extend to first order instead of propositional systems
  - More efficient inference methods

Expectation Maximization(EM)

- When to use:
  - Data is only partially observable
  - Unsupervised clustering (target value observable)
  - Supervised learning (Some instance attributes unobservable)
- Examples:
  - Train Bayesian belief networks
  - Unsupervised clustering (AUTOCLASS)
  - Learn hidden markov models

Generating Data from Mixture of k Gaussian Distributions

- Each instance x generated by
  - Choosing one of the k Gaussian with uniform distribution
  - Generating an instance at random according to that Gaussian



EM for Estimating k Means

Given:

- instances from X generated by mixture of k Gaussians
- unknown means $<\mu_1,\ldots,\mu_k>$ of the k Gaussians
- don't know which instance xi was generated by which Gaussian

Determine:

- maximum likelihood estimates of $<\mu_1,\ldots,\mu_k>$

Think of full description of each instance as $y_i=<x_i,z_{i1},z_{i2}>$

- z_j is 1 if xi generated by j-th Gaussian
- x_i observable
- z_j unobservable

EM for Estimating k Means

Think of full description of each instance as $y_i = \langle x_i, z_{i1}, z_{i2} \rangle$, where

- z_{ij} is 1 if x_i generated by jth Gaussian
- x_i observable
- z_{ij} unobservable

EM for Estimating k Means

EM algorithm: pick random initial $h=\langle \mu_1, \mu_2 \rangle$ then iterate

- E step: Calculate the expected value $E[z_{ij}]$ of each hidden variable zij, assuming the current hypothesis $h=\langle \mu_1, \mu_2 \rangle$ holds.

$$E[z_{ij}] = p(x=x_i|\mu=\mu_j) \,/\, \sum_{n=1}^{2} p(x=x_i|\mu=\mu_j)$$

$$= \exp(-(x_i-\mu_j)^2/2\sigma^2) \,/\, \sum_{n=1}^{2} \exp(-(x_i-\mu_n)^2/2\sigma^2)$$

- M step: Calculate a new maximum likelihood hypothesis $h'=\langle \mu_1', \mu_2' \rangle$ assuming the value taken on by each hidden variable z_{ij} is its expected value $E[z_j]$ calculated in the E - step. Replace $h=\langle \mu_1, \mu_2 \rangle$ by $h'=\langle \mu_1', \mu_2' \rangle$

$$\mu_j = \sum_{i=1}^{m} E[z_j]\, xi \,/\, \sum_{i=1}^{m} E[z_j]$$

EM Algorithm

- Converges to local maximum likelihood h and provides estimates of hidden variables zij
- In fact, local maximum in E[ln P(Y|h)]
- Y is complete (observable and unobservable variables) data
- Expected value is taken over possible values of unobserved variables in Y

General EM Problem

Given:

- observed data $X = \{x_1,\ldots,x_m\}$
- unobserved data $Z = \{z_1,\ldots,z_m\}$
- parameterized probability distribution $P(Y|h)$ where
  - $Y = \{y_1,\ldots,y_m\}$ is the full data $y_i=<x_i,z_i>$
  - h are the parameters

Determine:

- h that (locally) maximizes $E[\ln P(Y|h)]$

Applications:

- train Bayesian Belief Networks
- unsupervised clustering
- hidden Markov models

General EM Problem

Define likelihood function $Q(h'|h)$ which calculates $Y = X \cup Z$ using observed X and current parameters h to estimate Z

$$Q(h'|h) = E[\ \ln(P(Y|h')\ |\ h, X]$$

EM algorithm:

Estimation (E) step: Calculate $Q(h'|h)$ using the current hypothesis h and the observed data X to estimate the probability distribution over Y.

$$Q(h'|h) = E[\ \ln(P(Y|h')\ |\ h, X]$$

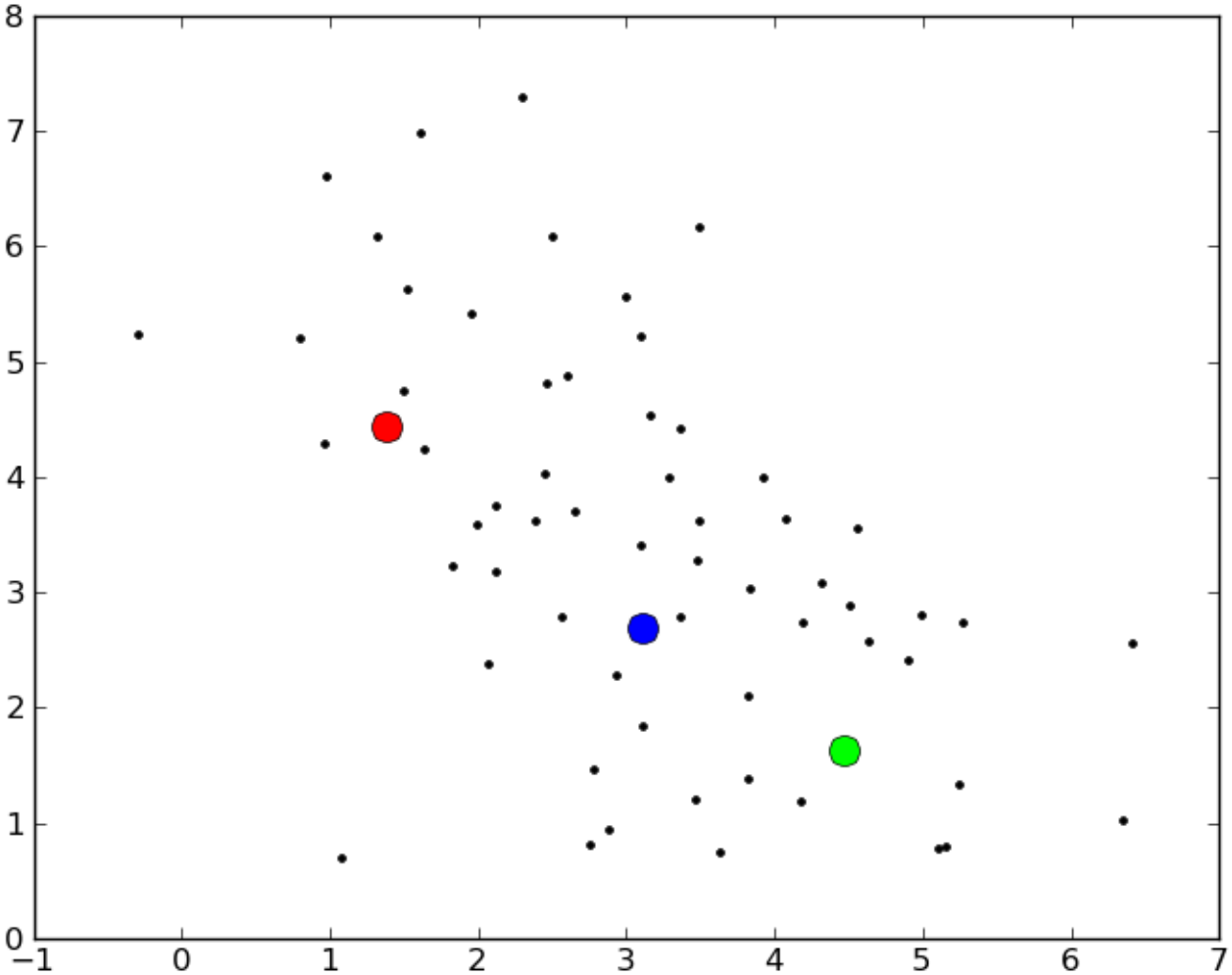
Maximization (M) step: Replace hypothesis h by the hypothesis h' that maximizes this Q function.

$$h = \text{argmax}_{h' \in H}\ Q(h'|h)$$

EM Example

- Implemented EM
- 3 Gaussian distributions
- Input
  - Samples drawn from the 3 distributions
  - Estimate of the mean of the 3 distributions

EM Example

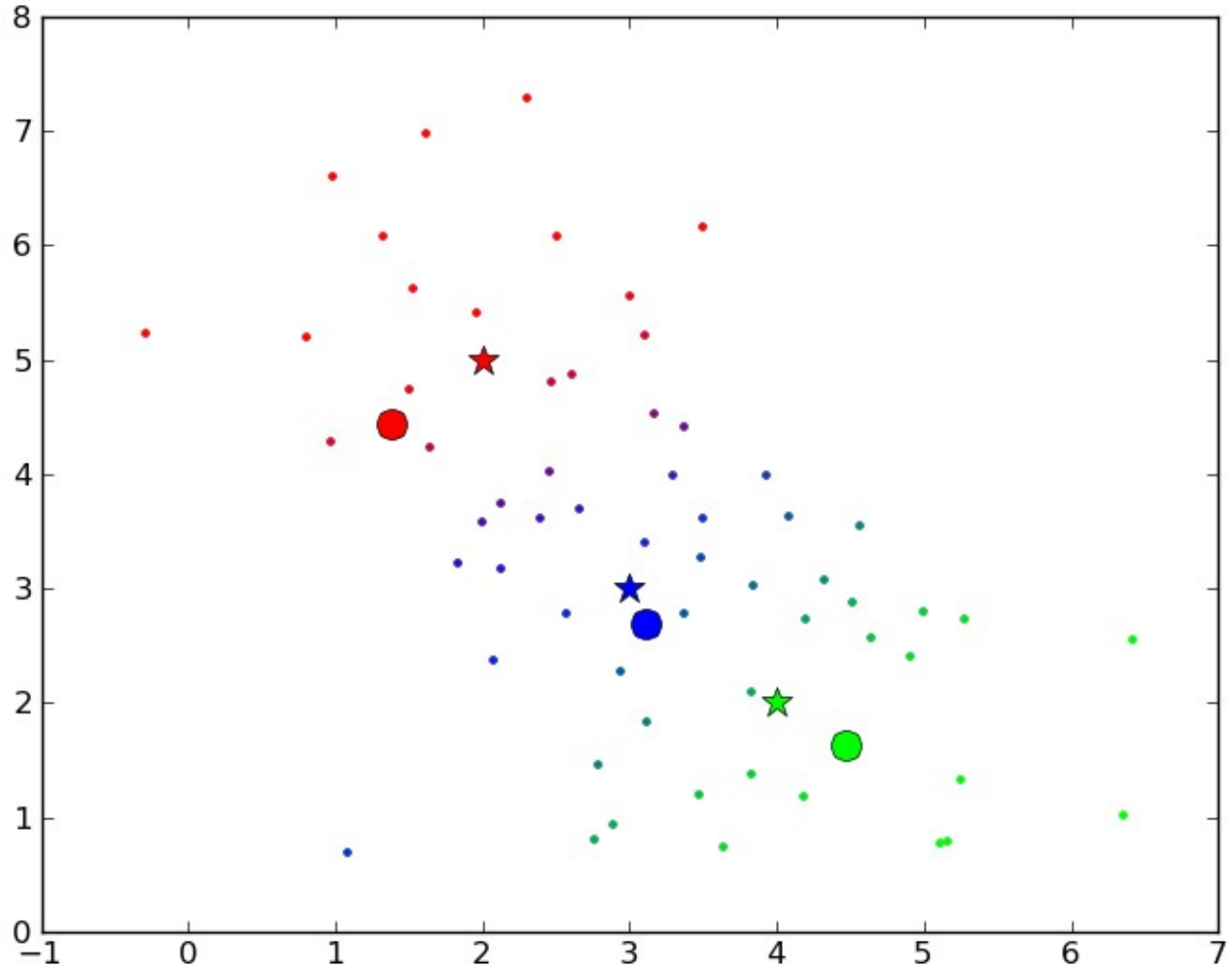


EM Example: Expectation Step

Estimated means
- circles

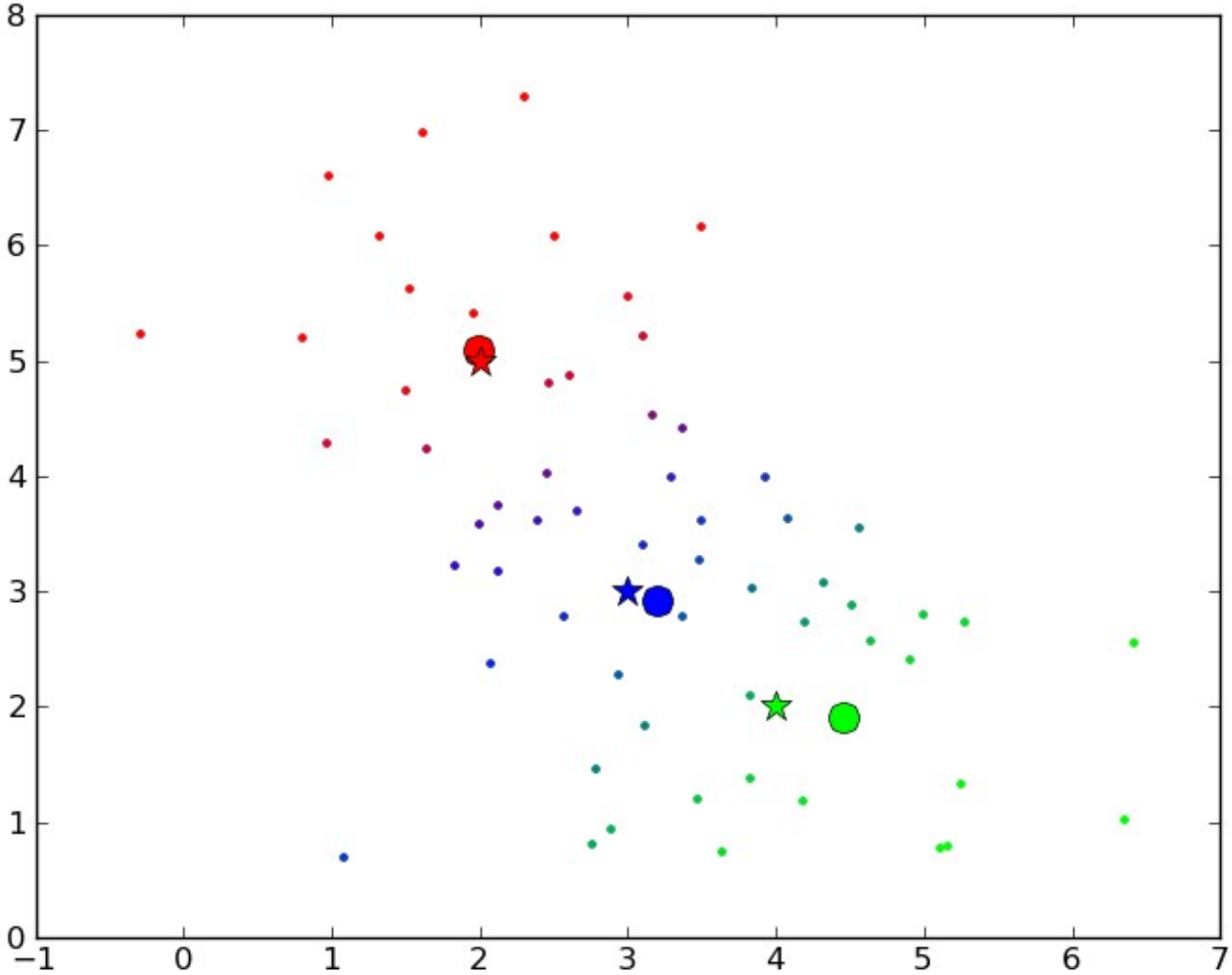
Real means
- stars

Sampled points are Colour coded for red, green, blue



EM Example: Maximization Step

- Calculate new centres based on weighted sum of inputs



Background

- http://plato.stanford.edu/entries/bayes-theorem/
- Frank Hoffman. http://www.nada.kth.se/kurser/kth/2D1431/02/index.html
- Work on probabilistic methods for mobile robot localization and mapping (SLAM) by Sebastian Thrun and Dieter Fox
- Expectation maximization