

---

## Chapter 6: Bayesian Learning (Part 2)

CS 536: Machine Learning  
Littman (Wu, TA)

## Administration

---

Terran Lane (U. New Mexico) agreed  
to be “Student Award Chair” for  
iCML.

November 12th, guest lecture.

Tenebaum talk?

*Schwarzenegger?*

## Bayesian Learning

---

[Read Ch. 6, except 6.3]

[Suggested exercises: 6.1, 6.2, 6.6]

- Bayes Theorem
- MAP, ML hypotheses
- MAP learners
- Minimum description length principle
- Bayes optimal classifier
- Naive Bayes learner
- Example: Learning over text data (today)
- Bayesian belief networks (today)
- Expectation Maximization algorithm (today)

## Learning to Classify Text

---

Why?

- Learn which news articles are of interest
  - Learn to classify web pages by topic
- Naive Bayes is among most effective algorithms
- What attributes shall we use to represent text documents??

## Learning to Classify Text

---

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

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

## Naïve Bayes for Text

---

- Naïve 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:

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

“Bag of words” assumption.

## Learning Algorithm

---

LEARN\_NAÏVE\_BAYES\_TEXT(*Examples*,  $V$ )

1. collect all words and other tokens that occur in *Examples*
  - *Vocabulary*  $\square$  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
    - $\text{docs}_j \square$  subset of *Examples* for which the target value is  $v_j$
    - $P(v_j) \square |\text{docs}_j|/|\text{Examples}|$
    - $\text{Text}_j \square$  a single document created by concatenating all members of  $\text{docs}_j$
    - $n \square$  total number of words in  $\text{Text}_j$  (counting duplicate words multiple times) (“tokens” vs. “tokens”)
    - for each word  $w_k$  in *Vocabulary*
      - $n_k \square$  number of times word  $w_k$  occurs in  $\text{Text}_j$
      - $P(w_k|v_j) \square (n_k + 1) / (n + |\text{Vocabulary}|)$

## Classification Algorithm

---

CLASSIFY\_NAÏVE\_BAYES\_TEXT (*Doc*)

- *positions*  $\square$  all word positions in *Doc* that contain tokens found in *Vocabulary*
- Return  $v_{NB}$ , where
$$v_{NB} = \operatorname{argmax}_{v_j \text{ in } V} P(v_j) \prod_{i \text{ in } \text{positions}} P(a_i|v_j)$$

## Twenty NewsGroups

---

Given 1000 training documents from each group, learn to classify new documents according to newsgroup:

comp.graphics	misc.forsale
comp.os.ms-windows.misc	rec.autos
comp.sys.ibm.pc.hardware	rec.motorcycles
comp.sys.mac.hardware	rec.sport.baseball
comp.windows.x	rec.sport.hockey
alt.atheism	sci.space
soc.religion.christian	sci.crypt
talk.religion.misc	sci.electronics
talk.politics.mideast	sci.med
talk.politics.misc	talk.politics.guns

Naïve Bayes: 89% classification accuracy

## Article in rec.sport.hockey

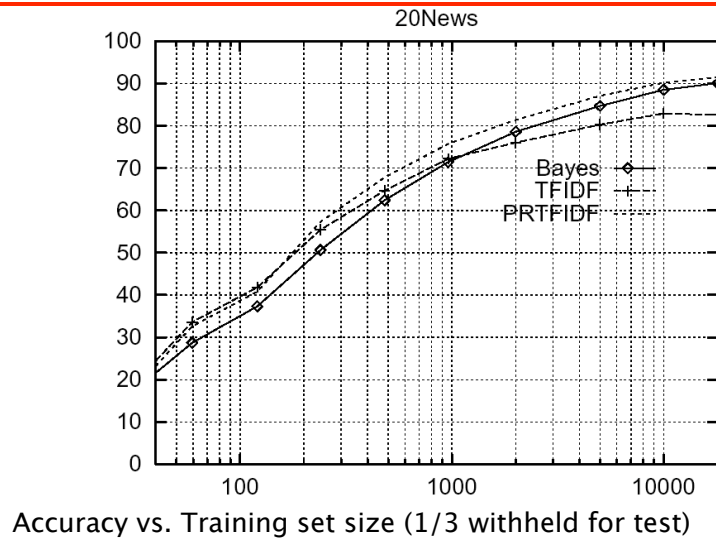
---

- Path: cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu
- From: xxx@yyy.zzz.edu (John Doe)
- Subject: Re: This year's biggest and worst (opinion)
- Date: 5 Apr 93 09:53:39 GMT

I can only comment on the Kings, but the most obvious candidate for pleasant surprise is Alex Zhitnik. He came highly touted as a defensive defenseman, but he's clearly much more than that. Great skater and hard shot (though wish he were more accurate). In fact, he pretty much allowed the Kings to trade away that huge defensive liability Paul Coffey. Kelly Hruddy is only the biggest disappointment if you thought he was any good to begin with. But, at best, he's only a mediocre goaltender. A better choice would be Tomas Sandstrom, though not through any fault of his own, but because some thugs in Toronto decided

## Learning Curve

---



## Bayesian Belief Networks

---

(Also called Bayes Nets, directed graphical models, BNs, ...). Interesting because:

- Naïve Bayes assumption of conditional independence too restrictive
- But it's intractable without some such assumptions...
- Belief networks describe conditional independence among subsets of variables
- allows combining prior knowledge about (in)dependencies among variables with observed training data

## 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

$$\begin{aligned} (\forall x_i, y_j, z_k) P(X = x_i | Y = y_j, Z = z_k) \\ = P(X = x_i | Z = z_k) \end{aligned}$$

more compactly, we write

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

## Independence Example

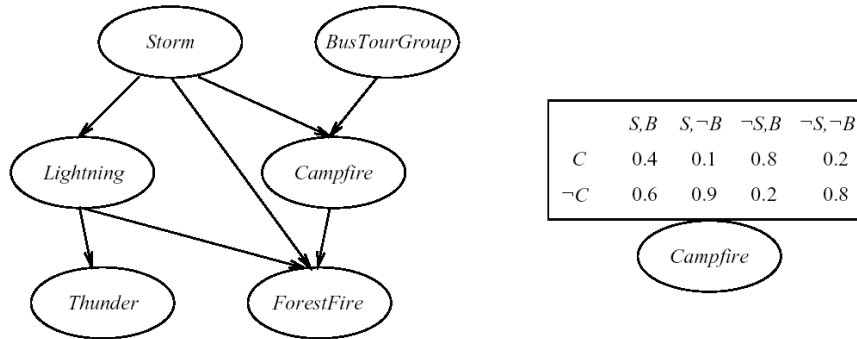
Example: *Thunder* is conditionally independent of *Storm*, given *Lightning*

$$\begin{aligned} P(\text{Thunder} | \text{Storm}, \text{Lightning}) \\ = P(\text{Thunder} | \text{Lightning}) \end{aligned}$$

Naive Bayes uses conditional ind. to justify

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

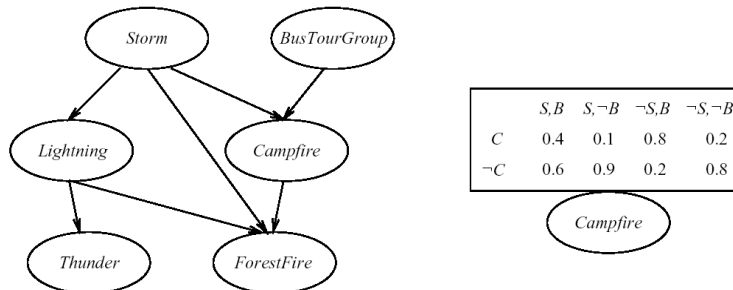
# Bayesian Belief Network



Network represents a set of conditional ind. assertions:

- Each node is asserted to be conditionally ind. of its nondescendants, given its immediate predecessors.
- Directed acyclic graph

# Bayesian Belief Network



Represents joint probability distribution over all variables

- e.g.,  $P(\text{Storm}, \text{BusTourGroup}, \dots, \text{ForestFire})$
- in general,  $P(y_1, \dots, y_n) = \prod_{i=1}^n P(y_i | \text{Parents}(Y_i))$   
where  $\text{Parents}(Y_i)$  denotes immediate predecessors of  $Y_i$  in graph
- so, joint distribution is fully defined by graph, plus the  $P(y_i | \text{Parents}(Y_i))$  (CPTs)

## Inference in Bayesian Nets

How can one infer the (probabilities of) values of one or more network variables, given observed values of others?

- Bayes net contains all information needed for this inference
- If only one variable with unknown value, easy to infer it
- Easy if BN is a “polytree”
- In general case, problem is NP hard (#P-complete, Roth 1996).

## Inference in Practice

In practice, can succeed in many cases

- Exact inference methods work well for some network structures (small “induced width”)
- Variational methods good approximation if nodes tightly coupled
- Monte Carlo methods “simulate” the network randomly to calculate approximate solutions

Now used as a primitive in more advanced learning and reasoning scenarios.

## Learning of Bayesian Nets

---

Several variants of this learning task

- Network structure might be *known* or *unknown*
- Training examples might provide values of *all* network variables, or just *some*

		structure known	unknown
observe	all	like Naïve Bayes	
	some		

## Learning Bayes Nets

---

Suppose structure known, variables partially observable

e.g., observe *ForestFire*, *Storm*, *BusTourGroup*, *Thunder*, but not *Lightning*, *Campfire*...

Similar to training neural network 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 BNs

---

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

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

e.g., if  $Y_i = \text{Campfire}$ , then  $u_{ik}$  might be  $\langle \text{Storm} = T, \text{BusTourGroup} = F \rangle$

Perform gradient ascent by repeatedly:

1. update all  $w_{ijk}$  using training data  $D$

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

2. then, renormalize the  $w_{ijk}$  to assure

$$\sum_j w_{ijk} = 1, 0 \leq w_{ijk} \leq 1$$

## More on Learning BNs

---

EM algorithm can also be used.

Repeatedly:

1. Calculate probabilities of unobserved variables, assuming  $h$

2. Calculate new  $w_{ijk}$  to maximize

$$E[\ln P(D|h)]$$

where  $D$  now includes both observed and (calculated probabilities of) unobserved variables

## Unknown Structure

---

When structure unknown...

- Algorithms use greedy search to add/subtract edges and nodes
- Active research topic

Somewhat like decision trees:  
searching for a discrete graph  
structure

## Summary: Belief Networks

---

- Combine prior knowledge with observed data
- Impact of prior knowledge (when correct!) is to lower the sample complexity
- Active research area (UAI)
  - Extend from Boolean to real-valued variables
  - Parameterized distributions instead of tables
  - Extend to first-order systems
  - More effective inference methods
  - ...

## Expectation Maximization

When to use EM:

- Data is only partially observable
- Unsupervised clustering (target value unobservable)
- Supervised learning (some instance attributes or labels unobservable)

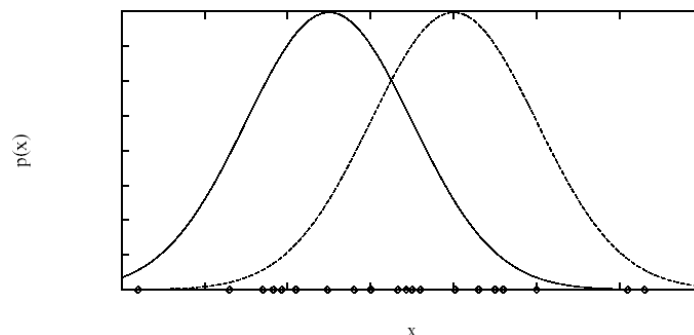
Some uses:

- Train Bayesian Belief Networks
- Unsupervised clustering (AUTOCLASS)
- Learning Hidden Markov Models

## Mixtures of $k$ Gaussians

Each instance  $x$  generated by

1. Choosing one of the  $k$  Gaussians with uniform probability
2. Generating an instance at random according to that Gaussian



## EM for Estimating $k$ Means

---

Given:

- Instances from  $X$  generated by mixture of  $k$  Gaussian distributions

Not given:

- Means  $\langle \mu_1, \dots, \mu_k \rangle$  of the  $k$  Gaussians
- Which instance  $x_i$  was generated by which Gaussian

Determine:

- ML estimates of  $\langle \mu_1, \dots, \mu_k \rangle$

## Missing Information

---

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  $j$ th Gaussian (not observed)
- $x_i$  observed data value

If we had full instances, how estimate  $\mu_1, \mu_2$ ?

If we had  $\mu_1, \mu_2$ , how predict  $z_{ij}$ ?

## 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  $z_{ij}$ , assuming the current hypothesis  $h = \langle \mu_1, \mu_2 \rangle$  holds.
- 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  $E[z_{ij}]$  calculated above. Replace  $h = \langle \mu_1, \mu_2 \rangle$  by  $h' = \langle \mu_1', \mu_2' \rangle$ .

## E Step For $k$ Means

---

$$E[z_{ij}] = \frac{p(x=x_i | \mu=\mu_j)}{\sum_{n=1}^k p(x=x_i | \mu=\mu_n)}$$

$$p(x=x_i | \mu=\mu_j) = \exp(-1/(2\sigma^2)(x_i - \mu_j)^2)$$

Derived via PDF for Gaussians and Bayes rule

## M Step For $k$ Means

---

$$\mu_j' = (\sum_{n=1}^m E[z_{ij}] x_i) / (\sum_{n=1}^m E[z_{ij}])$$

Estimates the mean by the sample mean (average of the observed data points, weighted by their probability of being generated by the Gaussian in question).

## EM Algorithm

---

Converges to local maximum likelihood  $h$  and provides estimates of hidden variables  $z_{ij}$

In fact, local maximum in  $E[\ln P(Y | h)]$

- $Y$  is complete (observable plus unobservable variables) data
- Expected value is taken over possible values of unobserved variables in  $Y$

## General EM Problem

---

Given:

- Observed data  $X = \{x_1, \dots, x_m\}$
- Unobserved data  $Z = \{z_1, \dots, z_m\}$
- Parameterized probability dist.  $P(Y | h)$ ,  
where
  - $Y = \{y_1, \dots, y_m\}$  is the full data  $y_i = x_i \cup z_i$
  - $h$  are the parameters

Determine:

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

## General EM Method

---

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

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

# Abstract Algorithm

---

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  $Y$ .

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

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

$$h \square \operatorname{argmax}_{h'} Q(h', h)$$