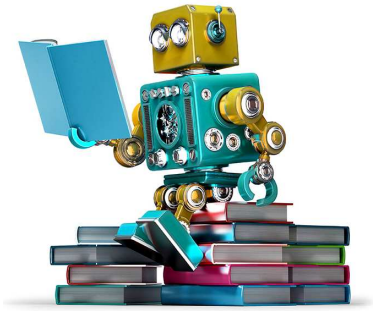
MACHINE LEARNING



MODULE-IV
BAYESIAN LEARNING

BY
HARIVINOD N
VIVEKANANDA COLLEGE OF ENGINEERING TECHNOLOGY, PUTTUR

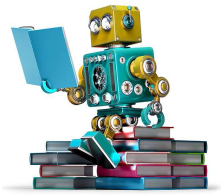
Hypothesis

- A hypothesis is a certain function that we believe (or hope) is similar to the true function, the *target function* that we want to model.
- In context of email spam classification, it would be the *rule* we came up with that allows us to separate spam from non-spam emails

Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Introduction

- probabilistic approach to inference
- **basic assumption**:
  - quantities of interest are governed by probability distributions
  - optimal decisions can be made by reasoning about these probabilities together with observed training data
- Bayesian Learning is relevant for two reasons
  - **first reason**: explicit manipulation of probabilities
    - among the most practical approaches to certain types of learning problems
    - e.g. Bayes classifier is competitive with decision tree and neural network learning

Introduction..(2)

MACHINE LEARNING

- Bayesian Learning is relevant for two reasons (cont.)
  - **second reason**: useful perspective for understanding learning methods that do not explicitly manipulate probabilites
    - determine conditions under which algorithms output the most probable hypothesis
    - e.g. justification of the error functions in ANNs
    - e.g. justification of the inductive bias of decision trees
- **features of Bayesian Learning methods:**
  - each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct
  - prior knowledge can be combined with observed data to determine the final probability of a hypothesis

Introduction..(3)

MACHINE LEARNING

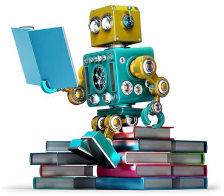
(e.g., hypotheses such as "this pneumonia patient has a 93% chance of complete recovery").

- **features of Bayesian Learning methods (cont.):**
  - hypotheses make probabilistaic predictions
  - new instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilites
  - standard of optimal decision making against which other practical measures can be measured
- **practical difficulties:**
  - initial knowledge of many probabilities is required
  - significant computational costs required

Module 4- Outline

Bayesian Learning

1. Introduction
2. **Bayes Theorem**
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Basics of Probability

- Prior probability
- Joint Probability
- Conditional Probability

- Example :Tossing 2 coins randomly……..
- P(Getting a tail)= ?
- P(Getting a head on first and head on second)=
- P(Getting a head on first given second is tail)=

Some definitions

MACHINE LEARNING

- *Product rule*: probability $P(A \wedge B)$ of a conjunction of two events A and B

$$P(A \wedge B) = P(A|B)P(B) = P(B|A)P(A)$$

- *Sum rule*: probability of a disjunction of two events A and B

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

- *Bayes theorem*: the posterior probability $P(h|D)$ of h given D

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

- *Theorem of total probability*: if events $A_1, \ldots, A_n$ are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1$, then

$$P(B) = \sum_{i=1}^{n} P(B|A_i)P(A_i)$$

Bayes Theorem

MACHINE LEARNING

- **machine learning** is interested in the *best hypothesis* h from some space H, given observed training data D
- *best hypothesis* $\approx$ *most probable hypothesis*
- Bayes Theorem provides a **direct method of calculating the probability of such a hypothesis** based on its prior probability, the probabilites of observing various data given the hypothesis, and the observed data itself

Bayes Theorem

- **more formal:**
  - $P(h)$ *prior probability of* h, reflects any background knowledge about the chance that h is correct
  - $P(D)$ *prior probability of* D, probability that D will be observed
  - $P(D|h)$ probability of observing D given a world in which h holds
  - $P(h|D)$ *posterior probability of* h, reflects confidence that h holds after D has been observed
- **Bayes Theorem:**

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

$$Posterior = \frac{Likelihood \times Prior}{Evidence}$$

Other Forms of Bayes Rule

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

$$Posterior = \frac{Likelihood \times Prior}{Evidence}$$

$$P(A|B) = \frac{P(B \mid A)P(A)}{P(B \mid A)P(A) + P(B \mid \sim A)P(\sim A)}$$

$$P(A|B \wedge X) = \frac{P(B \mid A \wedge X)P(A \wedge X)}{P(B \wedge X)}$$

MAP hypothesis

- in many learning scenarios, the learner considers some set of candidate hypotheses H and is interested in finding the most probable hypothesis $h \in H$ given the observed training data D
- any maximally probable hypothesis is called *maximum a posteriori* (**MAP**) hypotheses

$$\begin{aligned} h_{MAP} &= \underset{h \in H}{argmax}\ P(h|D) \\ &= \underset{h \in H}{argmax}\ \frac{P(D|h)P(h)}{P(D)} \\ &= \underset{h \in H}{argmax}\ P(D|h)P(h) \end{aligned}$$

note that $P(D)$ can be dropped, because it is a constant independent of h

ML Hypothesis

- sometimes it is assumed that every hypothesis is equally probable a priori
- in this case, the equation above can be simplified
- because $P(D|H)$ is often called the *likelihood of D given h*, any hypothesis that maximizes $P(D|h)$ is called *maximum likelihood* (**ML**) *hypothesis*

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

note that in this case $P(h)$ can be dropped, because it is equal for each $h \in H$

Example

MACHINE LEARNING

- consider a medical diagnosis problem in which there are two alternative hypotheses
  - the patient has a particular form of cancer (denoted by $cancer$)
  - the patient does not (denoted ny $\neg cancer$)
- the available data is from a particular laboratory with two possible outcomes: $\oplus$ (positive) and $\ominus$ (negative)

$$P(cancer) = .008 \qquad P(\neg cancer) = 0.992$$
$$P(\oplus|cancer) = .98 \qquad P(\ominus|cancer) = .02$$
$$P(\oplus|\neg cancer) = .03 \qquad P(\ominus|\neg cancer) = .97$$

- suppose a new patient is observed for whom the lab test returns a positive ($\oplus$) result
- Should we diagnose the patient as having cancer or not?

$$P(\oplus|cancer)P(cancer) = (.98).008 = .0078$$
$$P(\oplus|\neg cancer)P(\neg cancer) = (.03).992 = .0298$$
$$\Rightarrow h_{MAP} = \neg cancer$$

Example

MACHINE LEARNING

- the exact posterior probabilites can be determined by normalizing the above properties to 1

$$P(cancer|\oplus) = \frac{.0078}{.0078+0.0298} = .21$$
$$P(\neg cancer|\oplus) = \frac{.0298}{.0078+0.0298} = .79$$

$\Rightarrow$ the result of Bayesian inference depends strongly on the prior probabilities, which must be available in order to apply the method directly

Example-2

MACHINE LEARNING

Define A: has the disease B: test positive

We know:
P(A) = .001 $P(A^c)$ =.999
P(B|A) = .99 $P(B|A^c)$ =.02

We want to know P(A|B)=?

$$P(A \mid B) = \frac{P(A)P(B|A)}{P(A)P(B|A)+P(A^C)P(B|A^C)}$$

$$= \frac{.001 \times .99}{.001 \times .99 + .999 \times .02} = .0472$$

Example-3

MACHINE LEARNING

There is a 40% chance of it raining on Sunday. If it rains on Sunday, there is a 10% chance it will rain on Monday. If it didn't rain on Sunday, there's an 80% chance it will rain on Monday.

"Raining on Sunday" is event A, "Raining on Monday" is event B.

P(A) = 0.40 = Probability of Raining on Sunday.

P(A') = 0.60 = Probability of not raining on Sunday.

P(B|A) = 0.10 = Probability of it raining on Monday, if it rained on Sunday.

P(B'|A) = 0.90 = Probability of it not raining on Monday, if it rained on Sunday.

P(B|A') = 0.80 = Probability of it raining on Monday, if it did not rain on Sunday.

P(B'|A') = 0.20 = Probability of it not raining on Monday, if it did not rain on Sunday.

- What is the probability of it raining on Monday? - P(B)
- This would be the sum of the probability of "Raining on Sunday and raining on Monday" and "Not raining on Sunday and raining on Monday"

$$0.40 \times 0.10 + 0.60 \times 0.80 = 0.52 = 52\% \text{ chance}$$

Example-3…

"It rained on Monday. What is the probability it rained on Sunday?"

- This is where Bayes' theorem comes in.
- It allows us to calculate the probability of an earlier event, given the result of a later event.
- The equation used is: $P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$.
- P(B|A) = 0.10 = Probability of it raining on Monday, if it rained on Sunday.
- P(A) = 0.40 = Probability of Raining on Sunday.
- P(B) = 0.52 = Probability of Raining on Monday.
- So, to calculate the probability it rained on Sunday, given that it rained on Monday:

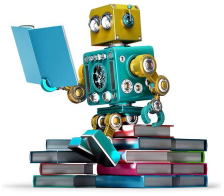
$$P(A|B) = \frac{0.10 * 0.40}{0.52} = .0769$$

i.e. if it rained on Monday, there's a 7.69% chance it rained on Sunday.

Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. **Bayes Theorem and Concept Learning**
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Bayes Theorem and Concept Learning

- What is the relationship between Bayes theorem and the problem of concept learning?
- it can be used for designing a straightforward learning algorithm
- **Brute-Force MAP LEARNING algorithm**
  1. For each hypothesis $h \in H$, calculate the posterior probability

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

  2. Output hypothesis h_{MAP} with the highest posterior probability

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

Brute-Force MAP Learning

- in order to specify a learning problem for the algorithm, values for $P(h)$ and $P(D|h)$ must be specified
- **assumptions**
  1. training data D is noise free (i.e., $d_i = c(x_i)$)
  2. target concept c is contained in H (i.e. $(\exists h \in H)[(\forall x \in X)[h(x) = c(x)]]$)
  3. no reason to believe that any hypothesis is more probable than any other

$$\Rightarrow P(h) = \frac{1}{|H|} \text{ for all } h \in H$$

$$\Rightarrow P(D|h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \text{ for all } d_i \in D \\ 0 & \text{otherwise} \end{cases}$$

Brute-Force MAP Learning..(2)

- now the problem for the learning algorithms is fully-defined
- in a first step, we have to determine the probabilites for $P(h|D)$
  - h is **inconsistent** with training data D

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0$$

  - h is **consistent** with training data D

$$P(h|D) = \frac{1 \cdot \frac{1}{|H|}}{P(D)} = \frac{1 \cdot \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1}{|VS_{H,D}|}$$

$\Rightarrow$ this analysis implies that, **under these assumptions**, each consistent hypothesis is a MAP hypothesis, because for each **consistent** hypothesis $P(h|D) = \frac{1}{|VS_{H,D}|}$

Brute-Force MAP learning..(3)

- Proof for derivation of P(D)

$$P(D) = \sum_{h_i \in H} P(D|h_i)\,P(h_i) = \sum_{h_i \in VS_{H,D}} 1 \cdot \frac{1}{|H|} + \sum_{h_i \notin VS_{H,\ D}} 0 \cdot \frac{1}{|H|}$$

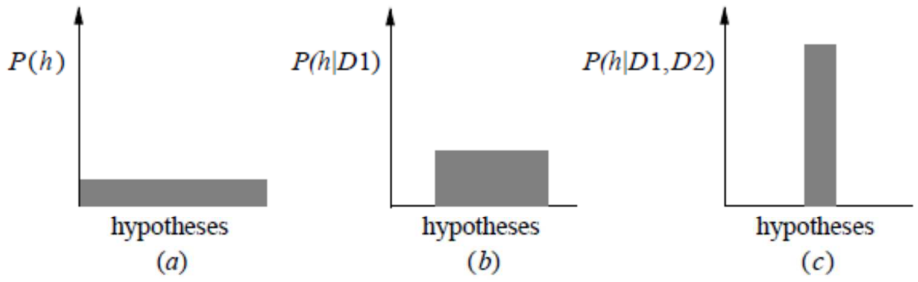
$$= \sum_{h_i \in VS_{H,D}} 1 \cdot \frac{1}{|H|} = \frac{|VS_{H,D}|}{|H|}$$

- To summarize, Bayes theorem implies that the posterior probability P(h|D) under our assumed P(h) and P(D|h) is

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

- Every consistent hypothesis is, therefore, a MAP hypothesis.

Brute-Force MAP Learning..(4)



- **evolution of probabilities**
  - **(a)** all hypotheses have the same probability
  - **(b)** + **(c)** as training data accumulates, the posterior probability of inconsistent hypotheses becomes zero while the total probability summing to 1 is shared equally among the remaining consistent hypotheses

Consistent Learner

- We will say that a learning algorithm is a **consistent learner** provided it outputs a hypothesis that commits zero errors over the training examples.
- Every consistent learner outputs a MAP hypothesis,
  - if we assume a uniform prior probability distribution over H (i.e., $P(h_i) = P(h_j)$ for all i, j), and
  - if we assume deterministic, noise free training data.

Consistent Learners

- FIND-S
  - outputs a consistent hypothesis and therefore a MAP hypothesis under the probability distributions $P(h)$ and $P(D|h)$ defined above
  - i.e. for each $P(h)$ that favors more specific hypotheses, FIND-S outputs a MAP hypotheses

⇒ Bayesian framework is a way to characterize the behaviour of learning algorithms

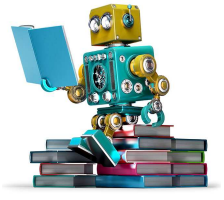
⇒ by identifying probability distributions $P(h)$ and $P(D|h)$ under which the output is a optimal hypothesis, implicit assumptions of the algorithm can be characterized (**Inductive Bias**)

⇒ inductive inference is modeled by an equivalent *probabilistic reasoning* system based on Bayes theorem

Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. **Maximum Likelihood and Least Square Hypothesis**
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Maximum Likelihood and Least-Squared Error

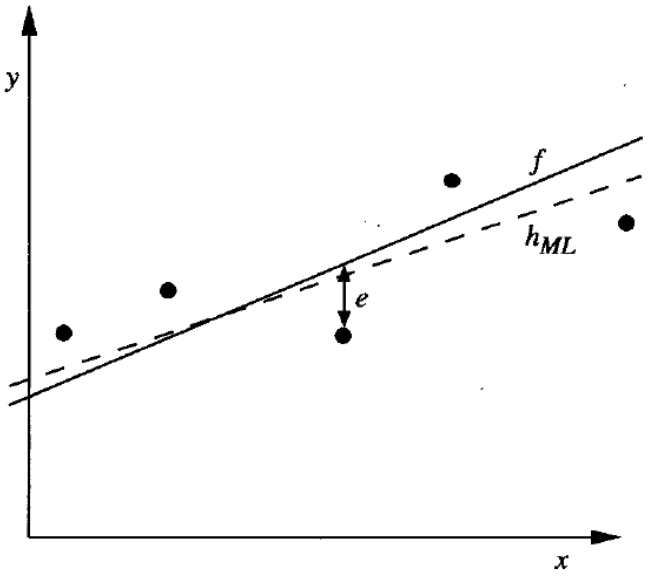
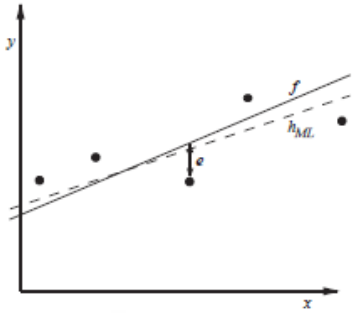


FIGURE 6.2
Learning a real-valued function. The target function f corresponds to the solid line. The training examples $\langle x_i, d_i \rangle$ are assumed to have Normally distributed noise e_i with zero mean added to the true target value $f(x_i)$. The dashed line corresponds to the linear function that minimizes the sum of squared errors. Therefore, it is the maximum likelihood hypothesis h_{ML}, given these five training examples.

Maximum Likelihood and Least-Squared Error



- **problem:** learning continuous-valued target functions (e.g. neural networks, linear regression, etc.)
- *under certain assumptions any learning algorithm that minimizes the squared error between the output hypothesis and the training data, will output a ML hypothesis*
- problem setting:
  - $(\forall h \in H)[h : X \rightarrow \Re]$ and training examples of the form $< x_i, d_i >$
  - unknown target function $f : X \rightarrow \Re$
  - m training examples, where the target value of each example is corrupted by random noise drawn according to a Normal probability distribution with zero mean $(d_i = f(x_i) + e_i)$

Maximum Likelihood and Least-Squared Error

MACHINE LEARNING

$$h_{ML} = \underset{h \in H}{argmax}\ p(D|h)$$

The training examples are assumed to be mutually independent given h.

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} p(d_i|h)$$

Given the noise e_i obeys a Normal distribution with zero mean and unknown variance σ^2, each d_i must also obey a Normal distribution around the true targetvalue $f(x_i)$.

Because we are writing the the expression for $P(D|h)$, we assume h is the correct description of f. Hence, $\mu = f(x_i) = h(x_i)$

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i - h(x_i))^2}$$

Maximum Likelihood and Least-Squared Error

MACHINE LEARNING

It is common to maximize the less complicated logarithm, which is justified because of the monotonicity of this function.

$$h_{ML} = \underset{h \in H}{argmax} \sum_{i=1}^{m} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

The first term in this expression is a constant independent of h and can therefore be discarded.

$$h_{ML} = \underset{h \in H}{argmax} \sum_{i=1}^{m} -\frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Maximizing this negative term is equivalent to minimizing the corresponding positive term.

$$h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} \frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Maximum Likelihood and Least-Squared Error

Finally, all constants independent of h can be discarded.

$$h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

$\Rightarrow$ the h_{ML} is one that minimizes the sum of the squared errors

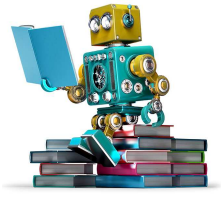
- Why is it reasonable to choose the Normal distribution to characterize noise?
  - good approximation of many types of noise in physical systems
  - Central Limit Theorem shows that the sum of a sufficiently large number of independent, identically distributed random variables itself obeys a Normal distribution

- only noise in the *target value* is considered, not in the *attributes describing the instances themselvs*

Module 4- Outline

Bayesian Learning

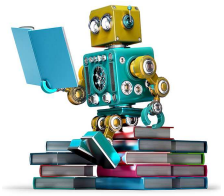
1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. **Maximum Likelihood Hypothesis for Predicting Probabilities**
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. **Minimum Description Length Principle**
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Minimum Description Length Principle

- recall **Occam's razor:** *choose the shortest explanation for the observed data*
- here, we consider a Bayesian perspective on this issue and a closely related principle
- **Minimum Description Length (MDL) Principle**
  - motivated by interpreting the definition of h_{MAP} in the light from information theory

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

$$h_{MAP} = \underset{h \in H}{argmax}\ \log_2 P(D|h) + \log_2 P(h)$$

$$h_{MAP} = \underset{h \in H}{argmin}\ -\log_2 P(D|h) - \log_2 P(h)$$

  - this equation can be interpreted as a statement that short hypotheses are preferred, assuming a particular representation scheme for encoding hypotheses and data

Minimum Description Length Principle

MACHINE LEARNING

- introduction to a basic result of information theory
  - consider the problem of designing a code C to transmit messages drawn at random
  - probability of encountering message i is p_i
  - interested in the most compact code C
  - Shannon and Weaver (1949) showed that the optimal code assigns $-\log_2 p_i$ bits to encode message i
  - $L_C(i) \approx$ *description length of message* i *with respect to* C
- $L_{C_H}(h) = -\log_2 P(h)$, where C_H is the optimal code for hypothesis space H
- $L_{C_{D|h}}(D|h) = -\log_2 P(D|h)$, where $C_{D|h}$ is the optimal code for describing data D assuming that both the sender and receiver know hypothesis h

$\Rightarrow$ **Minimum description length principle**

$$h_{MAP} = \underset{h \in H}{argmin}\, L_{C_H}(h) + L_{C_{D|h}}(D|h)$$

Minimum Description Length Principle

MACHINE LEARNING

- to apply this principle in practice, specific encodings or representations appropriate for the given learning task must be chosen
- **application to decision tree learning**
  - C_H might be some obvious encoding, in which the description length grows with the number of nodes and with the number of edges
  - choice of $C_{D|h}$?
    - sequence of instances $< x_1, ..., x_m >$ is known to the transmitter and the receiver
    - we need only to transmit the classifications $< f(x_1), ..., f(x_m) >$
    - if h correctly predicts the classification, no transmission is neccessary $(L_{C_{D|h}}(D|h) = 0)$
    - in case of misclassified examples, for each missclassification a message has to be sent that identifies this example (at most $\log_2 m$ bits) as well as its correct classification (at most $\log_2 k$ bits, where k is the number of possible classifications)

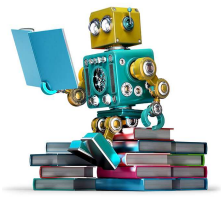
Minimum Description Length Principle

- MDL principle provides a way for trading off hypothesis complexity for the number of errors committed by the hypothesis
- It is one way of dealing with the issue of overfitting

Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. **Naïve Bayes Classifier**
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Naïve Bayesian Classifier

- applies to learning tasks where each instance x is described by a conjunction of attribute values and where the target function $f(x)$ can take on any value from some finite set V
- training examples are described by $< a_1, a_2, ..., a_n >$
- Bayesian approach

$$\begin{aligned} v_{MAP} &= \underset{v_j \in V}{argmax}\ P(v_j|a_1, a_2, ..., a_n) \\ &= \underset{v_j \in V}{argmax}\ \frac{P(a_1, a_2, ..., a_n|v_j)P(v_j)}{P(a_1, a_2, ..., a_n)} \\ &= \underset{v_j \in V}{argmax}\ P(a_1, a_2, ..., a_n|v_j)P(v_j) \end{aligned}$$

Naïve Bayesian Classifier

- $P(v_j)$ can be estimated by counting the frequency of v_j in D
- $P(a_1, a_2, ..., a_n|v_j)$ cannot by estimated in this fashion
  - number of these terms is $|X| \cdot |V|$
- **simplification of naive Bayes classifier**
  - attribute values are conditionally independent
  - hence, $P(a_1, a_2, ..., a_n|v_j) = \prod_i P(a_i|v_j)$
  - hence, number terms is |*distinct attributes*| · |*distinct target values*| + |*distrinct target values*|
  - no explicit search through H, just counting frequencies

$\Rightarrow$ **Naive Bayes Classifier**

$$v_{NB} = \underset{v_j \in V}{argmax}\ P(v_j) \prod_i P(a_i|v_j)$$

NBC-Illustrative Example

MACHINE LEARNING

Day	$Sunny$	$Temp.$	$Humidity$	$Wind$	$PlayTennis$
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

NBC-Illustrative Example

MACHINE LEARNING

- novel instance:

$$< Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong >$$

- Instantiation of the Naive Bayes Classifier

$$v_{NB} = \underset{v_j \in \{yes,no\}}{argmax} \; P(v_j) \prod_i P(a_i|v_j)$$

where

$$\prod_i P(a_i|v_j) = P(Outlook = sunny|v_j) \cdot P(Temperature = cool|v_j) \cdot P(Humidity = high|v_j) \cdot P(Wind = strong|v_j)$$

- estimation of probabilities

$$P(PlayTennis = yes) = \frac{9}{14} = .64 \; P(PlayTennis = no) = \frac{5}{14} = .36$$

NBC-Illustrative Example

MACHINE LEARNING

- similarly, conditional probabilites can be estimated (e.g. $Wind = Strong$)

$$P(Wind = Strong|PlayTennis = yes) = \frac{3}{9} = .33$$
$$P(Wind = Strong|PlayTennis = no) = \frac{3}{5} = .60$$

- calculation of v_{VB}

$$P(yes)P(sunny|yes)P(cool|yes)P(high|yes)P(strong|yes) = .0053$$
$$P(no)P(sunny|no)P(cool|no)P(high|no)P(strong|no) = .0206$$
$$\Rightarrow v_{VB} = no$$

- normalization

$$\frac{.0206}{.0206+.0053} = .795$$

Estimating probabilities

MACHINE LEARNING

- normally, probabilites are estimated by the fraction of times the event is observed to occur over the total number of opportunities ($\frac{n_c}{n}$)
- in most cases, this method is a good estimate
- but if n_c is very small, it provides poor results
  - biased underestimate of the probability
  - if this estimate equals zero, it will dominate the Bayes classifier
- Bayesian approach: ***m-estimate***

$$\frac{n_c + mp}{n + m}$$

where p is a prior estimate of the probability we wish to determine, and m is a constant called the *equivalent sample size* which determines how heavily to weight p relative to the observed data

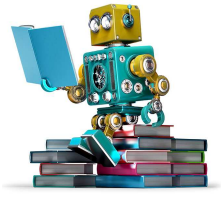
Estimating probabilities

- in the absence of information, it is common to assume a uniform distribution for p
- hence, $p = \frac{1}{k}$ where k is the number of possible attribute values
- if $m = 0$, the ***m-estimate*** is equivalent to $\frac{n_c}{n}$
- m can be interpreted as the number of virtual samples distributed according to p that are added the n actual observed examples

Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. **Bayesian Belief Networks**
9. EM Algorithm
10. Summary

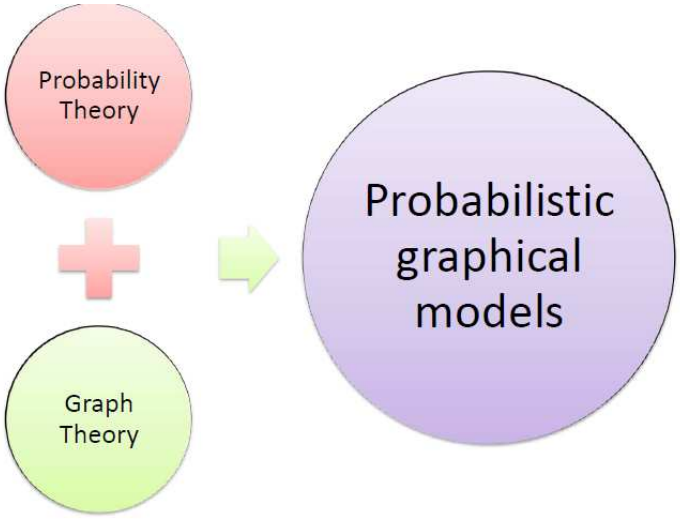


Bayesian Belief Networks

MACHINE LEARNING

- **motivation**
  - naive Bayes classifier makes significant use of the assumption of conditional independence
  - this assumption dramatically reduces the complexity of the learning task
  - however, in many cases this assumption is overly restrictive
- **Bayesian Belief Network**
  - describes probability distribution governing a set of variables by specifying a **set of conditional independence assumptions** along with a **set of conditional probabilites**
  - conditional independence assumption applies only to subsets of the variables

MACHINE LEARNING

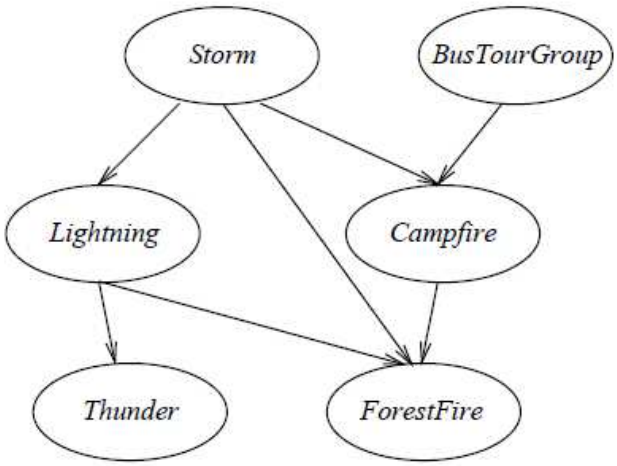


Notation

- Bayesian Belief Networks describe the probability distribution over a set of variables
- arbitrary set of random variables $Y_1, ..., Y_n$ where $V(Y_i)$ is the set of possible values for Y_i
- *joint space*: $V(Y_1) \times V(Y_2) \times ... \times V(Y_n)$
- *joint probability distribution* specifies the probability for each of the possible variable bindings for the tuple $< Y_1, Y_2, ..., Y_n >$

Representation

Bayesian networks (BN) are represented by directed acyclic graphs.



	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

Campfire

Representation

- joint probability distribution over the boolean variables $Storm$, $Lighting$, $Thunder$, $ForestFire$, $CampFire$, and $BusTourGroup$
- set of conditional independence assumptions
  - represented by a DAG
  - node ≈ variables in the joint space
  - arcs ≈ conditional dependence of the originator
- for each node a conditional probability table is given
  - describes probability distribution for the variable given the values of its immediate precedessors
  - the joint probability for any desired assignment of $< y_1, y_2, ..., y_n >$ is computed by

$$P(y_1, y_2, ..., y_n) = \prod_{i=1}^{n} P(y_i | Parents(Y_i))$$

where $Parents(Y_i)$ denotes the set of immediate precedessors of Y_i

Inference

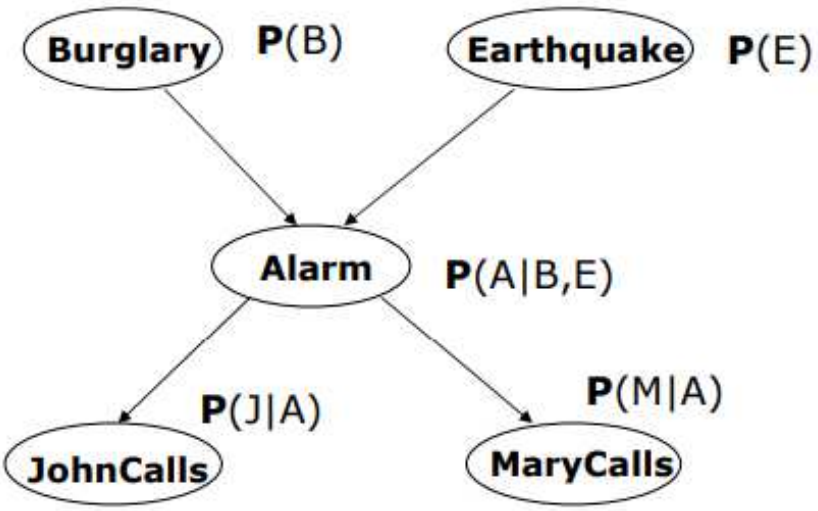
- **task:** inference of the probability distribution for a target value, e.g. $ForestFire$
- if values are known for all other variables in the network, the inference is straightforward
- in the more general case, values are only known for a subset of the network variables
- a Bayesian Network can be used to compute the the probability distribution for any subset of network variables given the values or distributions for any subsetof the remaining variables
- exact inference is NP-hard, even approximate inference can be NP-hard

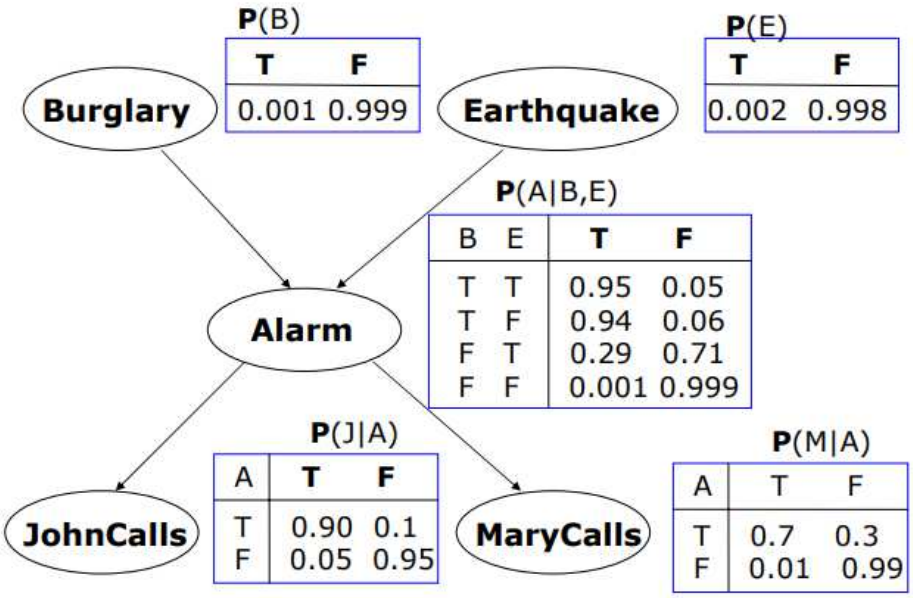
Example

I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?

Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*
Network topology reflects "causal" knowledge:

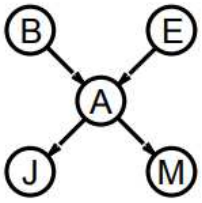
- A burglar can set the alarm off
- An earthquake can set the alarm off
- The alarm can cause Mary to call
- The alarm can cause John to call





Compactness

A CPT for Boolean X_i with k Boolean parents has 2^k rows for the combinations of parent values



Each row requires one number p for $X_i = true$ (the number for $X_i = false$ is just $1 - p$)

If each variable has no more than k parents, the complete network requires $O(n \cdot 2^k)$ numbers

I.e., grows linearly with n, vs. $O(2^n)$ for the full joint distribution

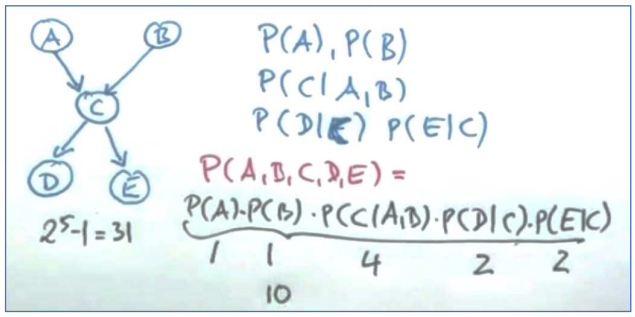
For burglary net, $1 + 1 + 4 + 2 + 2 = 10$ numbers (vs. $2^5 - 1 = 31$)

Conditional Probability Table(CPT)

A Joint Distribution for a network with n Boolean nodes has $2^n - 1$ rows for the combinations of parent values.

B	E	A	J	M	P()
1	1	1	1	1	?
1	1	1	1	0	?
1	1	1	0	1	?
1	1	1	0	0	?
1	1	0	1	1	?
1	1	0	1	0	?
:	:	:	:	:	:
:	:	:	:	:	:
0	0	0	0	0	?

Total: 32 rows... Ok, 31.

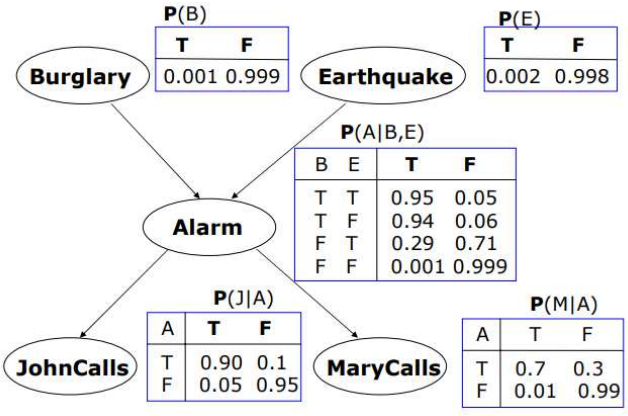


"Global" semantics defines the full joint distribution as the product of the local conditional distributions:

$$P(x_1, \ldots, x_n) = \Pi_{i=1}^{n} P(x_i | parents(X_i))$$

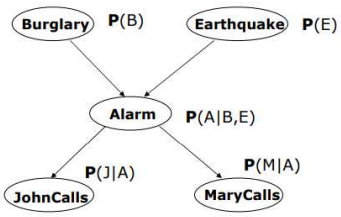
e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

$$= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e)$$
$$= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998$$
$$\approx 0.00063$$



I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?
Choose:

$$Goal : argmax_{b \in \{t,f\}} P(j = t, m = t, a = t, b, e = f)$$



$$P(j = t, m = t, a = t, b = t, e = f) = P(j = t|a = t)P(m = t|a = t)P(a = t|b = t, e = f)P(b = t)P(e = f)$$

vs

$$P(j = t, m = t, a = t, b = f, e = f) = P(j = t|a = t)P(m = t|a = t)P(a = t|b = t, e = f)P(b = f)P(e = f)$$

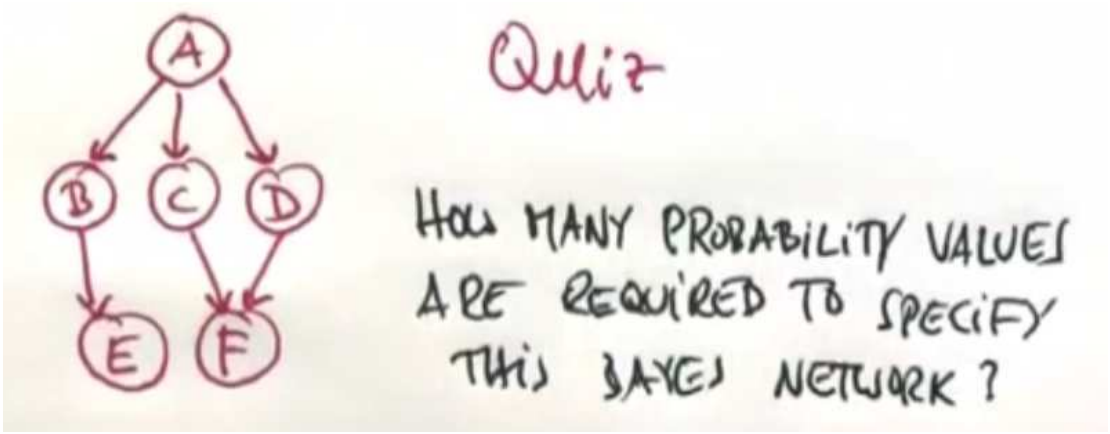
.000628

Categorizations of Algorithms

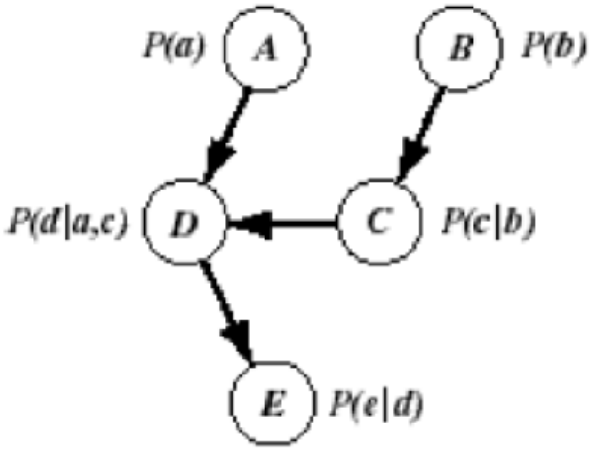
MACHINE LEARNING

- **network structure**: known or unknown
- **network variables**: observable or partially observable
- in case of known structure and fully observable variables, the conditional probabilities can be estimated as for naive Bayes classifier
- in case of known structure and partially observable variables, the learning problem can be compared to learning weights for an ANN (Russel et al., 1995)
- in case of unknown stucture, heuristic algorithms or scoring metric have to be used (Cooper and Herskovits, 1992)

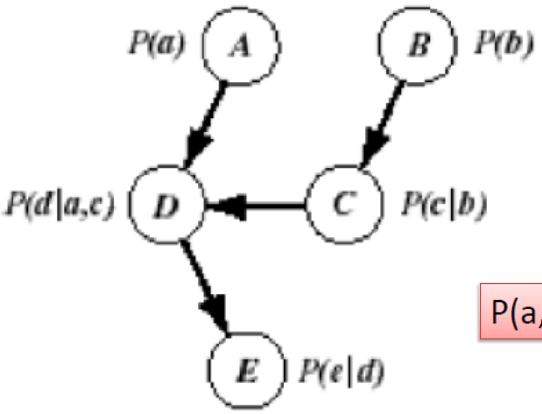
Quiz



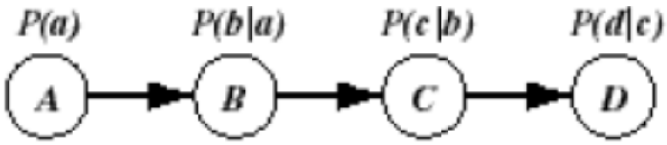
Quiz



$$P(a, b, c, d, e) = P(a)P(b)P(c|b)P(d|a,c)P(e|d)$$



$$P(a, b, c, d, e) = P(a)P(b)P(c|b)P(d|a,c)P(e|d)$$

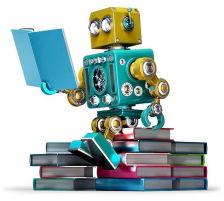


$$P(a, b, c, d) = P(a)P(b|a)P(c|b)P(d|c)$$

Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. **EM Algorithm**
10. Summary



Motivation

- In many practical learning settings, only a subset of the relevant instance features might be observable.
- For example, among many *Storm, Lightning, Thunder, ForestFire, Campfire,* and *BusTourGroup* have been observed. (In BBN example)
- If some variable is sometimes observed and sometimes not, then we can use the cases for which it has been observed to learn to predict its values when it is not.
- Many approaches have been proposed to handle the problem of learning in the presence of unobserved variables.
- EM algorithm (Dempster et al. 1977), a widely used approach to learning in the presence of unobserved variables.
- The EM algorithm can be used
  - even for variables whose value is never directly observed,
  - provided the general form of the probability distribution governing these variables is known.

Estimating Means of k Gaussians

- Consider a problem in which the data D is a set of instances are - a mixture of k distinct Normal distributions.
- This problem setting is illustrated in Figure for the case where k = 2 and where the instances are the points shown along the x axis.
- Each instance is generated using a two-step process.
  - First, one of the k Normal distributions is selected at random.
  - Second, a single random instance x_i is generated according to this selected distribution.
- This process is repeated to generate a set of data points as shown in the figure.

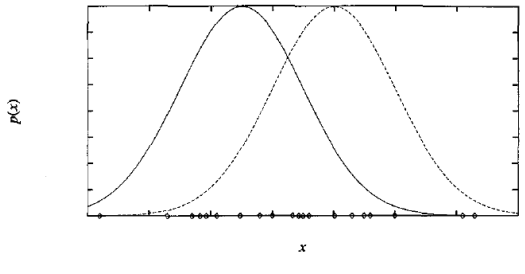
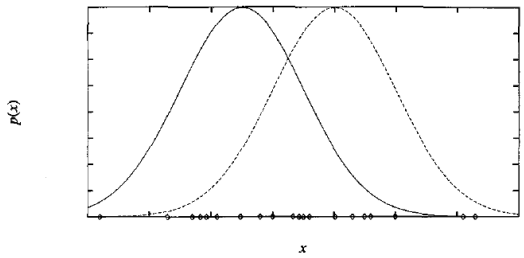


FIGURE 6.4
Instances generated by a mixture of two Normal distributions with identical variance σ. The instances are shown by the points along the x axis. If the means of the Normal distributions are unknown, the EM algorithm can be used to search for their maximum likelihood estimates.

Estimating Means of k Gaussians

MACHINE LEARNING

- To simplify our discussion, we consider the special case
  - where the selection of the single Normal distribution at each step is based on choosing each with uniform probability,
  - where each of the k Normal distributions has the same variance σ^2, known value.
- The learning task is to output a hypothesis h = (μ_1, . . . ,μ_k) that describes the means of each of the k distributions.
- We would like to find a maximum likelihood hypothesis for these means; that is, a hypothesis h that maximizes p(D |h).



$$\mu_{ML} = \underset{\mu}{\operatorname{argmin}} \sum_{i=1}^{m} (x_i - \mu)^2$$

FIGURE 6.4
Instances generated by a mixture of two Normal distributions with identical variance σ. The instances are shown by the points along the x axis. If the means of the Normal distributions are unknown, the EM algorithm can be used to search for their maximum likelihood estimates.

Estimating Means of k Gaussians

MACHINE LEARNING

- Our problem here, however, involves a mixture of k different Normal distributions, and we cannot observe which instances were generated by which distribution.
- we can think full description of each instance as the triple **(x_i, z_{i1}, z_{i2})**,
  - where x_i is the observed value of the i^{th} instance and
  - where z_{i1} and z_{i2} indicate which of the two Normal distributions was used to generate the value x_i.
- In particular, z_{ij} has the value 1 if x_i was created by the j^{th} Normal distribution and 0 otherwise.
- Here x_i is the observed variable in the description of the instance, and z_{il} and z_{i2} are hidden variables.
  - If the values of z_{il} and z_{i2} were observed, we could use following Equation to solve for the means p_1 and p_2.
  - Because they are not, we will instead use the EM algorithm.

EM algorithm

Step 1: 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.

Step 2: 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 in Step 1. Then replace the hypothesis $h = \langle \mu_1, \mu_2 \rangle$ by the new hypothesis $h' = \langle \mu'_1, \mu'_2 \rangle$ and iterate.

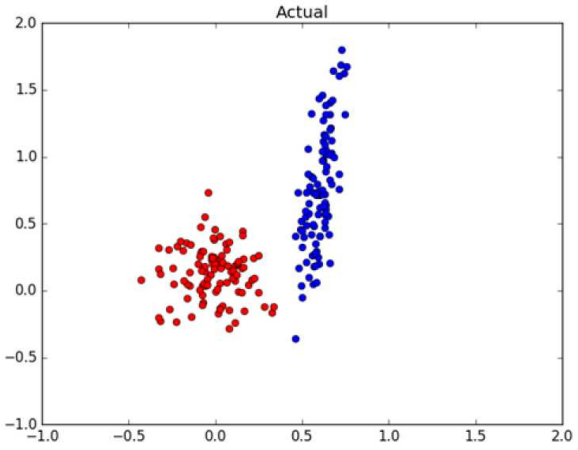
Step 1

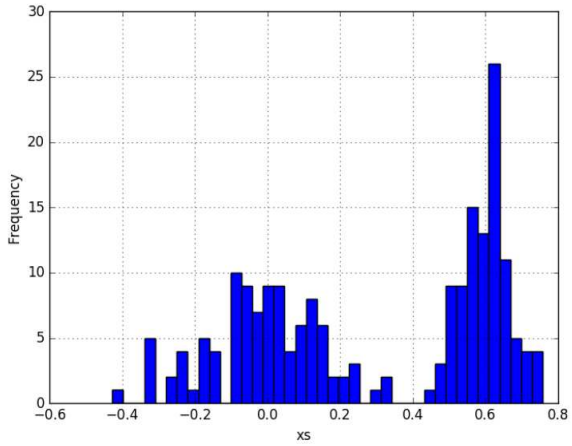
$$E[z_{ij}] = \frac{p(x = x_i | \mu = \mu_j)}{\sum_{n=1}^{2} p(x = x_i | \mu = \mu_n)} = \frac{e^{-\frac{1}{2\sigma^2}(x_i - \mu_j)^2}}{\sum_{n=1}^{2} e^{-\frac{1}{2\sigma^2}(x_i - \mu_n)^2}}$$

Step 2

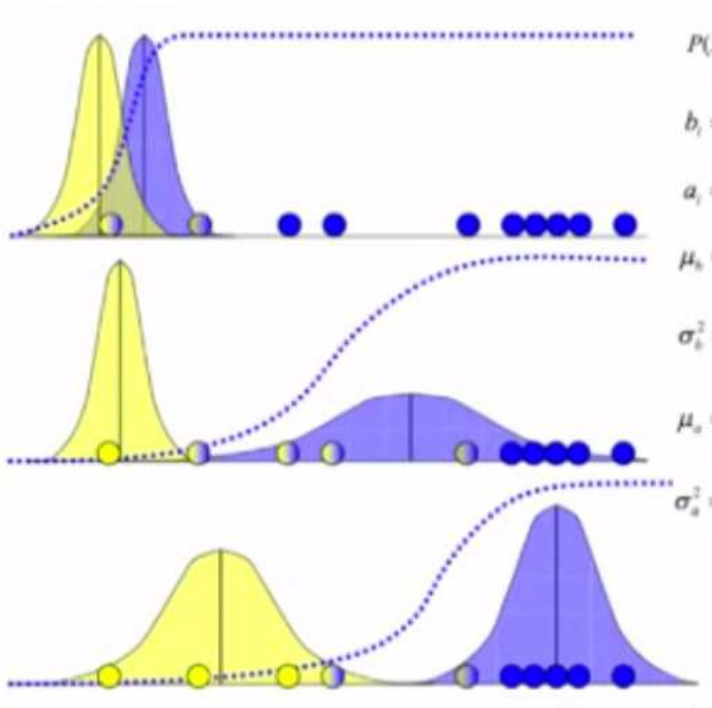
$$\mu_j \leftarrow \frac{\sum_{i=1}^{m} E[z_{ij}] \; x_i}{\sum_{i=1}^{m} E[z_{ij}]}$$

The current hypothesis is used to estimate the unobserved variables, and the expected values of these variables are then used to calculate an improved hypothesis.





EM 1-d example



$$P(x_i \mid b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{(x_i - \mu_b)^2}{2\sigma_b^2}\right)$$

$$b_i = P(b \mid x_i) = \frac{P(x_i \mid b)P(b)}{P(x_i \mid b)P(b) + P(x_i \mid a)P(a)}$$

$$a_i = P(a \mid x_i) = 1 - b_i$$

$$\mu_b = \frac{b_1 x_1 + b_2 x_2 + \ldots + b_n x_n}{b_1 + b_2 + \ldots + b_n}$$

$$\sigma_b^2 = \frac{b_1 (x_1 - \mu_1)^2 + \ldots + b_n (x_n - \mu_n)^2}{b_1 + b_2 + \ldots + b_n}$$

$$\mu_a = \frac{a_1 x_1 + a_2 x_2 + \ldots + a_n x_n}{a_1 + a_2 + \ldots + a_n}$$

$$\sigma_a^2 = \frac{a_1 (x_1 - \mu_1)^2 + \ldots + a_n (x_n - \mu_n)^2}{a_1 + a_2 + \ldots + a_n}$$

could also estimate priors:

$P(b) = (b_1 + b_2 + \ldots b_n) / n$

$P(a) = 1 - P(b)$

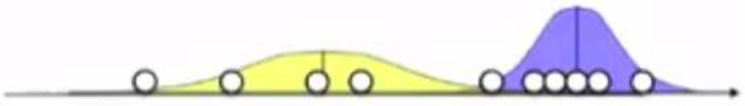
Mixture models in 1D

- Observations $x_1 \ldots x_n$
  - K=2 Gaussians with unknown μ, σ^2
  - estimation trivial if we know the source of each observation

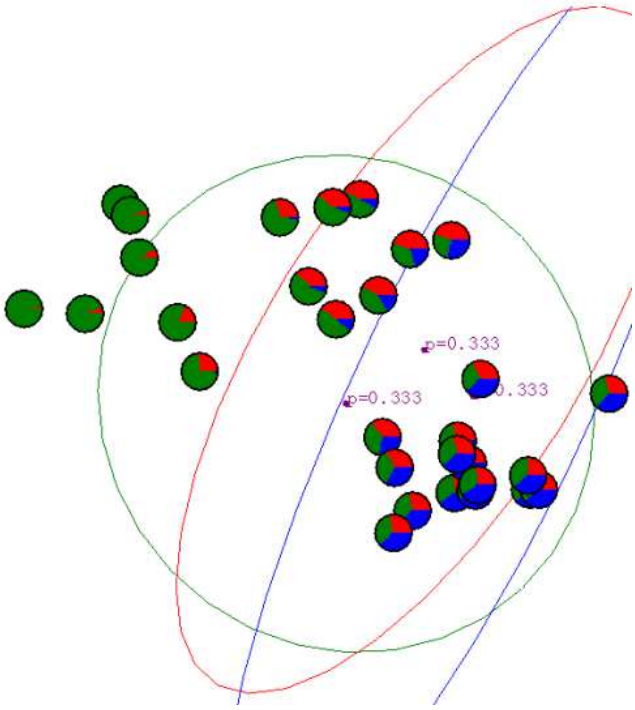
$$\mu_b = \frac{x_1 + x_2 + \ldots + x_{n_b}}{n_b}$$

$$\sigma_b^2 = \frac{(x_1 - \mu_1)^2 + \ldots + (x_n - \mu_n)^2}{n_b}$$

- What if we don't know the source?
- If we knew parameters of the Gaussians (μ, σ^2)
  - can guess whether point is more likely to be a or b

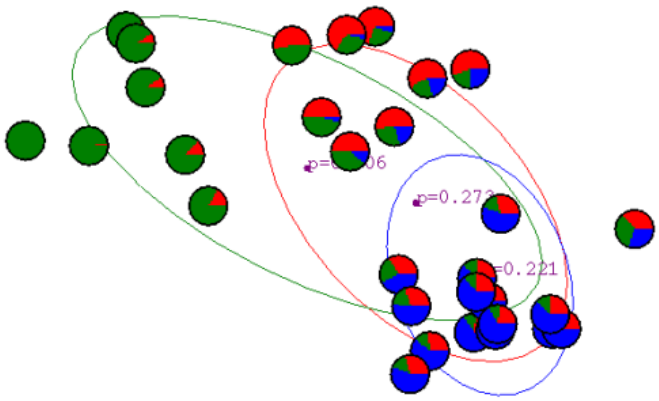


Gaussian Mixture Example: Start

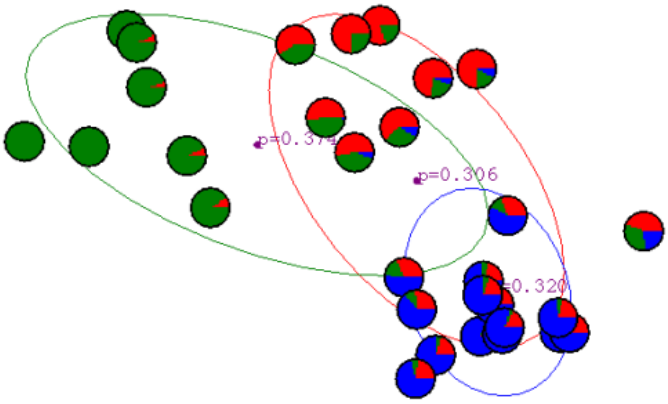


Advance apologies: in Black and White this example will be incomprehensible

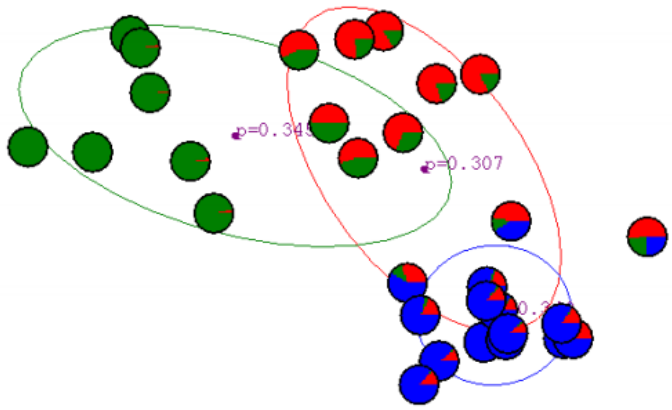
After first iteration



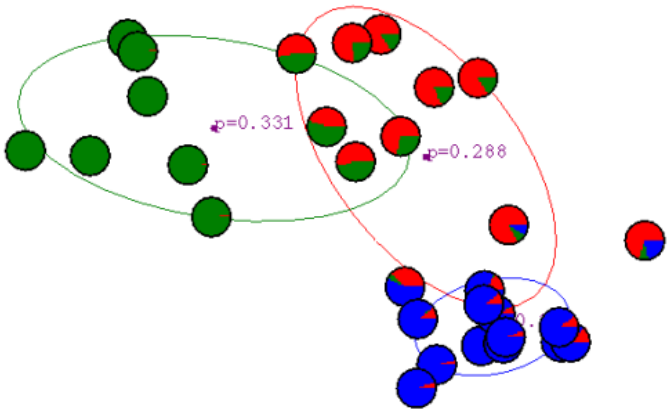
After 2nd iteration



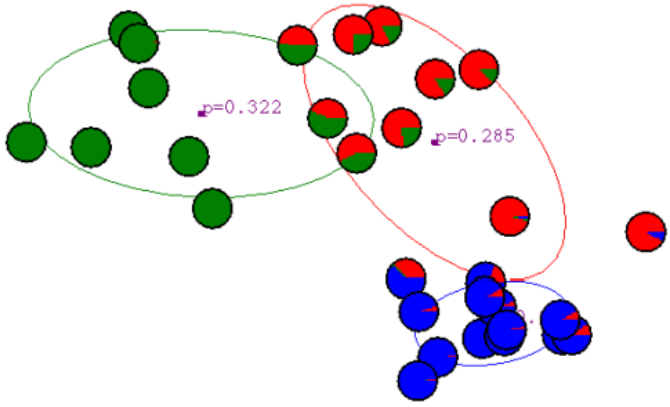
After 3rd iteration



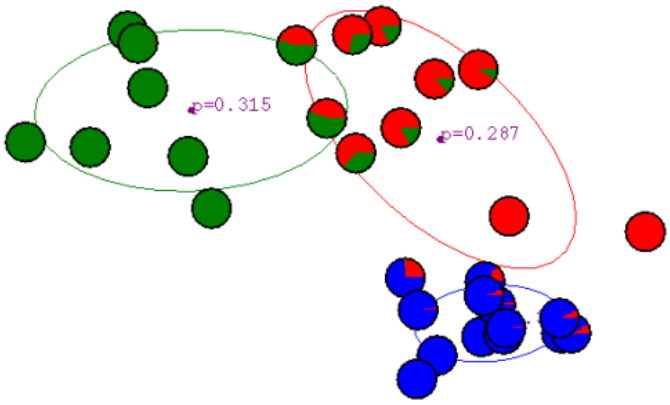
After 4th iteration



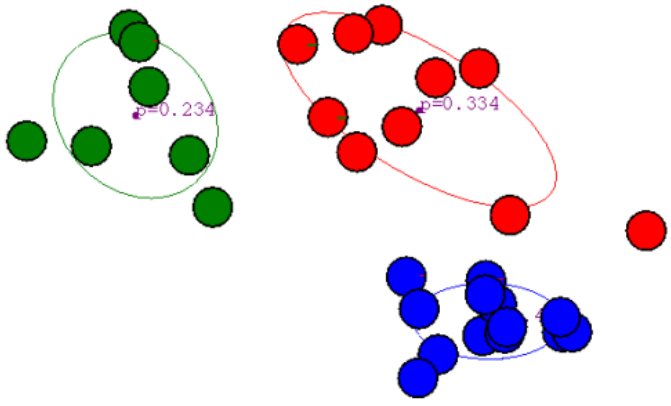
After 5th iteration



After 6th iteration



After 20th iteration



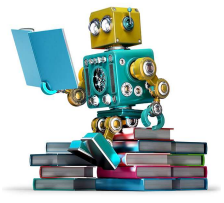
EM Algorithm

- Chicken and egg problem
  - need (μ_a, σ_a^2) and (μ_b, σ_b^2) to guess source of points
  - need to know source to estimate (μ_a, σ_a^2) and (μ_b, σ_b^2)
- EM algorithm
  - start with two randomly placed Gaussians (μ_a, σ_a^2), (μ_b, σ_b^2)
  - for each point: $P(b|x_i)$ = does it look like it came from b?
  - adjust (μ_a, σ_a^2) and (μ_b, σ_b^2) to fit points assigned to them
  - iterate until convergence

Module 4- Outline

Bayesian Learning

1. Introduction
2. Bayes Theorem
3. Bayes Theorem and Concept Learning
4. Maximum Likelihood and Least Square Hypothesis
5. Maximum Likelihood Hypothesis for Predicting Probabilities
6. Minimum Description Length Principle
7. Naïve Bayes Classifier
8. Bayesian Belief Networks
9. EM Algorithm
10. Summary



Summary

- Bayesian methods provide the basis for probabilistic learning methods that accommodate (and require) knowledge about the prior probabilities of alternative hypotheses and about the probability of observing various data given the hypothesis.
- Bayesian methods allow assigning a posterior probability to each candidate hypothesis, based on these assumed priors and the observed data.
- Bayesian methods can be used to determine the most probable hypothesis given the data-the maximum a posteriori (MAP) hypothesis.
  - This is the optimal hypothesis in the sense that no other hypothesis is more likely.

Summary

- The framework of Bayesian reasoning can provide a useful basis for analyzing certain learning methods that do not directly apply Bayes theorem.
  - For example, under certain conditions it can be shown that minimizing the squared error when learning a real-valued target function corresponds to computing the maximum likelihood hypothesis.
- The Minimum Description Length principle recommends choosing the hypothesis that minimizes the description length of the hypothesis plus the description length of the data given the hypothesis.
  - Bayes theorem and basic results from information theory can be used to provide a rationale for this principle.

Summary

- The naive Bayes classifier is a Bayesian learning method that has been found to be useful in many practical applications.
- It is called "naive" because it incorporates the simplifying assumption that attribute values are conditionally independent, given the classification of the instance.
- When this assumption is met, the naive Bayes classifier outputs the MAP classification.
- Even when this assumption is not met, as in the case of learning to classify text, the naive Bayes classifier is often quite effective.
- Bayesian belief networks provide a more expressive representation for sets of conditional independence assumptions among subsets of the attributes.

Summary

- In many practical learning tasks, some of the relevant instance variables may be unobservable.
- The EM algorithm provides a quite general approach to learning in the presence of unobservable variables.
  - This algorithm begins with an arbitrary initial hypothesis.
  - It then repeatedly calculates the expected values of the hidden variables (assuming the current hypothesis is correct), and then recalculates the maximum likelihood hypothesis (assuming the hidden variables have the expected values calculated by the first step).
- This procedure converges to a local maximum likelihood hypothesis, along with estimated values for the hidden variables.