Bayesian Learning

Burr H. Settles
CS-540, UW-Madison
www.cs.wisc.edu/~cs540-1
Summer 2003

Announcements (7/23)

- Grades to date (including grades for HW#2) are on the class website
  - "Late days" means the number of late days that have *been used* so far this semester
- Homework #5 is still in the works…

Announcements (7/24)

- Homework #5 is finally done!
  - **Good news:** due date for HW4 and HW5 are extended to Tuesday, 7/29
  - **Bad news:** no late days (apologies if you have been saving them up!)
- TA's solution to HW2 is on the website as well if you're still interested in improving your Mancala playing agent

Learning With Probabilities

- Sometimes machine learning algorithms are too rigid or brittle to be applied in many real-world problems
  - e.g. How would decision trees or *k*-NN fare in stochastic, dynamic, or partially observable environments?
- Sometimes the agent should make decisions based on what is most likely to happen, or what the world is likely to actually be like
  - The agent can then use its experience to learn these sorts of probabilities
- But let's take a break from talking about learning for a moment, and introduce what it means to be Bayesian

Thomas Bayes (1702-1761)

- Thomas Bayes was a Nonconformist Presbyterian minister in England in the early 18th century
- Described by William Wiston as *"… a dissenting Minister… and a successor, though not immediate, to Mr. Humphrey Ditton, and like him a very good mathematician."*

Thomas Bayes

- Bayes was elected a Fellow of the *Royal Society of London* (sort the M.I.T. of the day) in 1742
  - Despite the fact that at that time he had no published works on mathematics to his name
  - He actually had published one *anonymously*: a critique of George Berkeley's attack on the logic of probability
- Apparently Bayes tried to retire from the chapel in Tunbridge Wells (where he was minister) in 1749
  - Perhaps to focus more on his math "hobby"
  - But didn't actually retire until 1752
  - Stayed in Tunbridge Wells until his death in 1761

Thomas Bayes

- In 1764, Bayes' friend Richard Price found one of his papers, "Essay towards solving a problem in the doctrine of chances" and submitted it to the Royal Society
  - Bayes' first scientific publication: printed in the *Philosophical Transactions of the Royal Society of London*, 3 years after he died
  - The principles in this paper were accepted by Pierre-Simon Laplace (of differential equation fame) and Marquis de Condorcet (infinitesimals) later in the on century

✱ *Bayes' findings are the basis for much of modern probability theory (certainly the parts that we use in AI, anyway)*

7

Probability Theory

- A Random variable (RV): a variable that takes on values from a set of mutually exclusive and exhaustive values
- $A=a$: a proposition, variable A has a particular value a
  - This can correspond to a percept or feature, e.g. *Wind=Weak*
- $P(A=a)$: single probability of RV $A=a$, which is the degree of belief in a proposition in the absence of any other relevant information
  - e.g. $P(Wind=Weak)$, etc.
- $P(A)$: probability distribution, i.e. set of $P(A=a_i)$ for all i
  - e.g. $P(Wind) = \{ P(Wind=Weak), P(Wind=Strong) \}$

8

Probability Theory

- Joint probabilities specify the probabilities for the conjunction of propositions
  - e.g. $P(A,B)$ or $P(A \wedge B)$
- A full joint probability distribution:
  - Completely specifies all of the possible probabilities by enumerating all possible variable-value combinations
  - Kind of like a truth table
  - Intractable representation: since table grows exponentially in size k^n where n variables each have k possible values

9

Probability Theory

- Conditional (posterior) probabilities:
  - Formalize the process of accumulating evidence and updating probabilities based on new evidence
  - Specify the belief in one proposition (event, conclusion, diagnosis, etc.) conditioned on another proposition (evidence, feature, symptom, etc.)
- $P(A \mid B)$ is the conditional probability of A given evidence B is known to be true

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

10

Probability Theory

- Conditional probabilities behave like standard probabilities:
  - e.g. $0 \leq P(A|B) \leq 1$
    - Conditional probabilities are within the range 0 to 1, inclusive
  - $P(A=a_1 \mid B) + P(A=a_2 \mid B) + ... + P(A=a_n \mid B) = 1$
    - Conditional probabilities sum to 1
- Can have $P(\textit{conjunction of events} \mid B)$
  - $P(A \wedge B \wedge C \mid E)$ is the agent's belief in the sentence "$A \wedge B \wedge C$" conditioned on the evidence E being true

11

Independence

- Unconditional (absolute) Independence: variables that have no connection to each other
  - Taking CS-540 has no relationship to having a hayfever
    - $P(cs540 \mid hayfever) = P(cs540)$
    - $P(hayfever \mid cs540) = P(hayfever)$
    - $P(cs540 \wedge hayfever) = P(cs540) \times P(hayfever)$
- Conditional Independence: variables that are connected only through another variable
  - Sneezing and drowsiness connected to hayfever, but not each other
    - $P(sneeze \mid drowsy \wedge hayfever) = P(sneeze \mid hayfever)$
    - $P(drowsy \mid sneeze \wedge hayfever) = P(drowsy \mid hayfever)$
    - $P(sneeze \wedge drowsy \mid hayfever) = P(sneeze \mid hayfever) \times P(drowsy \mid hayfever)$

12

Rules of Probability Theory

Negation
Probability event A being false:

$$P(\neg A \mid B) = 1 - P(A \mid B)$$

Sum Rule
Probability of a disjunction of two events *A* and *B*:

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

Product Rule
Probability of a conjunction of two events *A* and *B*:

$$P(A \wedge B) = P(A \mid B) \times P(B) = P(B \mid A) \times P(A)$$

13

Rules of Probability Theory

Chain Rule
Generalization of the product rule for any number of events:

$$P(A \wedge B \wedge C) = P(A \mid B \wedge C) \times P(B \mid C) \times P(C)$$

Conditional Chain Rule
Variant of the chain rule for conditional probabilities:

$$P(A \wedge B \mid C) = P(A \mid B \wedge C) \times P(B \mid C)$$

Total Probability
"Summing out" over mutually exclusive events $B_1,\ldots,B_n$

$$P(A) = \sum_{i=1}^{n} P(A \mid B_i) \times P(B_i)$$

14

Bayes' Rule

Bayes' Rule is the basis for efficiently computing unknown conditional probabilities, as derived from the product rule:

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

$$P(A \mid B) \times P(B) = P(B \mid A) \times P(A)$$

$$\boxed{P(A \mid B) = \frac{P(B \mid A) \times P(A)}{P(B)}}$$

15

Bayes' Rule Examples

Bayes' Rule: $P(A \mid B) = \dfrac{P(B \mid A) \times P(A)}{P(B)}$

***P*(*happy* | *sunny*) = 0.95**
***P*(*sunny*) = 0.5**
***P*(*happy*) = 0.75**
***P*(*sunny* | *happy*) =**
(0.95 × 0.5)/0.75 =
0.63

***P*(*sneeze* | *cold*) = 0.75**
***P*(*cold*) = 0.1**
***P*(*sneeze*) = 0.2**
***P*(*cold* | *sneeze*) =**
(0.75 × 0.1)/0.2 =
0.375

16

Relationship to Machine Learning

- For inductive learning, we are given a training set *D* of examples, from which we are to approximate a concept function *f*
- We have a set of hypotheses *H* (i.e. the hypothesis space), from which we want to choose a single hypothesis *h* such that $h \approx f$

✱ *Then it makes sense to choose the most probable hypothesis* h∈ H *given the training data* D

17

Choosing Hypotheses

Using Bayes' Rule, we can measure the probability of a hypothesis *h* given evidence of the training data *D*:

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

Generally, we want to find the most probable hypothesis $h \in H$, called the maximum a posteriori hypothesis h_{MAP}:

$$h_{MAP} = \arg\max_{h \in H} P(h \mid D) = \arg\max_{h \in H} \frac{P(D \mid h) \times P(h)}{P(D)} = \arg\max_{h \in H} P(D \mid h) \times P(h)$$

If we assume $P(h_i) = P(h_j)$, all hypotheses are equally probable, and we can further simplify to the maximum likelihood hypothesis h_{ML}:

$$h_{ML} = \arg\max_{h_i \in H} P(D \mid h_i)$$

18

Bayesian Learning

- Bayesian learning is, generally speaking, the method of selecting the best hypothesis $h \in H$ in terms of how well it can explain the observed training data D:
  – If hypotheses have different probabilities:

$$h_{MAP} = \arg\max_{h \in H} P(D \mid h) \times P(h)$$

  – If hypotheses are equally likely:

$$h_{ML} = \arg\max_{h_i \in H} P(D \mid h_i)$$

Surprise Candy!

- Consider this example from p.712 of the textbook:
  – A surprise candy comes in two flavors (cherry, lime)
  – There are 5 kinds of unmarked candy bags:

h_1 = 100% cherry	$P(h_1) = 0.1$
h_2 = 75% cherry, 25% lime	$P(h_2) = 0.2$
h_3 = 50% cherry, 50% lime	$P(h_3) = 0.4$
h_4 = 25% cherry, 75% lime	$P(h_4) = 0.2$
h_5 = 100% lime	$P(h_5) = 0.1$

- Since you love cherry but hate lime, you want to hypothesize about (e.g. learn) which bag you have

Surprise Candy!

- At this point, you believe that the bag is most likely h_3 (50% cherry, 50% lime), because it has this highest probability $P(h_3) = 0.4$
- However, as you take the candies out to examine them (i.e. collect training data), the most probable hypothesis will change
  – Since each hypothesis has a different probability (some bags are more common than others), we want to find the h_{MAP} hypothesis
  – Each h_i is scored by $P(D \mid h_i) \times P(h_i)$, where:

$$P(D \mid h_i) = \prod_{d \in D} P(d \mid h_i)$$

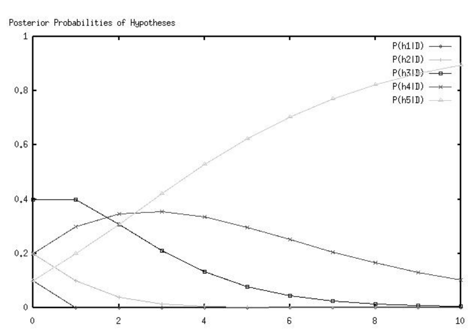
Surprise Candy!

- After examining 10 candies that are *all lime*, the probabilities for each bag (hypothesis) are:

$P(h_1 \mid D)$	$= (0.0)^{10} \times 0.1$	$= 0$
$P(h_2 \mid D)$	$= (0.25)^{10} \times 0.2$	$= 2\times10^{-7}$
$P(h_3 \mid D)$	$= (0.5)^{10} \times 0.4$	$= 4\times10^{-4}$
$P(h_4 \mid D)$	$= (0.75)^{10} \times 0.2$	$= 0.01$
$P(h_5 \mid D)$	$= (1.0)^{10} \times 0.1$	$= 0.1$

- It should be intuitively obvious to us that, after 10 lime candies in a row, this is an all-lime bag, but now we can give the agent a similar intuition!

Surprise Candy!



Bayesian Learning in Practice

- Consider our inductive learning framework, specifically inducing decision trees:
  – Hypothesis space H = set of all possible decision trees for the problem
  – A training set D (assume that it is noise-free)

∗ *Does ID3 find a MAP hypothesis?*

Decision Trees and MAP

- Recall the definition of a *MAP* hypothesis:

$$h_{MAP} = \arg\max_{h \in H} P(D \mid h) \times P(h)$$

- Since D is assumed to be noise-free, any tree h that we induce (regardless of heuristic) will be fully consistent with D, thus $P(D \mid h) = 1.0$
- So the key factor is $P(h)$… but how can we measure that?

25

Probability of Hypotheses

- Interesting fact from information theory: the optimal code (i.e. shortest expected "coding length") for an event with probability p is $-\log_2 p$ bits…

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

26

Minimum Description Length

$$h_{MAP} = \arg\min_{h \in H} -\log_2 P(D \mid h) - \log_2 P(h)$$

- We can interpret:
  - $-\log_2 P(h)$ to be the "size" of hypothesis h
  - $-\log_2 P(D \mid h)$ to be the number of h's misclassifications in D (e.g. "impurity" of predictions on D given hypothesis h)
- We call this the the minimum description length principle
  - We want to trade-off hypothesis size (complexity) for accuracy
- For a decision tree h and noise-free training set D, then $-\log_2 P(D \mid h) = 0$, so we want the want the smallest tree h
  - A theoretical defense for Occam's Razor!

27

Bayesian Classification

- So far, we've just talked about finding the most probable hypothesis h given data D
  - Boils down to Occam's razor: doesn't teach us anything new!
- Given a test example x, what is its most probable classification? Is it necessarily $h_{MAP}(x)$?
- Consider:
  - Four hypotheses
    - $P(h_1 \mid D) = .4$ $P(h_2 \mid D)=.2$ $P(h_3 \mid D)=.2$ $P(h_4 \mid D)=.2$
  - And a new instance x
    - $h_1(x)$=YES $h_2(x)$=NO $h_3(x)$= NO $h_4(x)$= NO
  - What's the most probable label for example x?

28

Bayesian Classification

- It turns out that we can also use Bayes' Rule to do inductive learning
  - Consider input features $x_1,\ldots,x_n$ to be evidence
  - Probability of a class label c is $P(c \mid x_1,\ldots,x_n)$
- There are many different inductive learning algorithms that use Bayesian probability theory to predict the most likely class label
  - Naïve Bayes Classifier
  - Bayesian Belief Network (Bayes Net)

    We'll talk about these
  - Bayes Optimal Classifier
  - Gibbs Classifier

29

Naïve Bayes Classifier

- Along side decision trees, k-NN, and neural nets, a naïve Bayes classifier one of the most practical and widely-used inductive learning algorithms
- Naïve Bayes classifiers are extremely fast:
  - Training scales linearly with respect to $|D|$ (number of training examples)
  - Testing is linear in $|x|$ (number of features)
- Makes the naïve Bayes assumption:
  - All features (evidence) are independent of each other
  - In other words, if one feature should be conditioned on another, the classifier is "naïve" (unaware of it)

30

Naïve Bayes Classifier

- A concept function $f : X \rightarrow C$, where $c \in C$ is a class label, and $x \in X$ is described by a feature vector $\langle x_1,\dots,x_n \rangle$:

$$
\begin{aligned}
c_{MAP} &= \arg\max_{c \in C} P(c \mid x_1,\dots,x_n) \\
&= \arg\max_{c \in C} \frac{P(x_1,\dots,x_n \mid c) \times P(c)}{P(x_1,\dots,x_n)} \\
&= \arg\max_{c \in C} P(x_1,\dots,x_n \mid c) \times P(c)
\end{aligned}
$$

- The Naïve Bayes assumption is that all the features are independent (e.g. *Sunny* has nothing to do with *Windy*)

$$c_{NB} = \arg\max_{c \in C} P(c) \times \prod_n P(x_n \mid c)$$

31

Naïve Bayes Algorithm

- To learn from training set D:
  - For each concept class c
    - $PE(c) \leftarrow$ estimation of $P(c)$ according to D
    - For each input feature observation x_n
      - $PE(x_n \mid c) \leftarrow$ estimation of $P(x_n \mid c)$ according to D
- Then, to classify an new example x:
  - Score each c by: $PE(c) \times \prod_n [PE(x_n \mid c)]$
  - Return the c with the highest probability

32

Naïve Bayes Example

- Recall our "circus" example from a few lectures ago…
- Let's use a Naïve Bayes classifier to predict if >1,000 people will attend the circus based on the following weather forecast:
  - $x = \langle$*Outlook=Sunny*, *Temp=Cool*, *Humid=High*, *Wind=Strong*$\rangle$
- So we want to compute:

$$c_{NB} = \arg\max_{c \in C} P(c) \times \prod_n P(x_n \mid c)$$

$$P(Y) \times P(Sunny \mid Y) \times P(Cool \mid Y) \times P(High \mid Y) \times P(Strong \mid Y) = .005$$

$$P(N) \times P(Sunny \mid N) \times P(Cool \mid N) \times P(High \mid N) \times P(Strong \mid N) = .021$$

$$\rightarrow c_{NB} = No$$

33

Issues With Naïve Bayes

- In practice, we estimate the probabilities by maintaining counts as we pass through the training data, and then divide through at the end
- But what happens if, when classifying, we come across a feature-value we didn't see in training (e.g. *Temperature=Sub-Zero*)?

$$PE(x_n \mid c) = 0 \text{ … therefore…}$$

$$PE(c) \times \prod_n PE(x_n \mid c) = 0$$

- Typically, we can get around this by initializing all the counts to Laplacian priors (small uniform values) instead of 0
  - This way, the probability will still be small, but not impossible
  - This is also called "smoothing"

34

Issues With Naïve Bayes

- In order to use naïve Bayes classification, our features need to be a set of events/propositions on which to condition the probability estimates
  - This is straightforward for Boolean features
  - For discrete features, we can do what we did with neural nets: enumerate all the feature-value pairs
- But what about continuous features?
  - The only thing we can do is generate new Boolean features from the older continuous ones
  - Could use the method we used for splitting in decision trees
  - More commonly, we want a wider discretized range, so we make k equally-sized "bins" to represent continuous value ranges

35

Issues With Naïve Bayes

- Similar to the problems with continuous features, notice that we call this a naïve Bayes "classifier"
  - Not a "naïve Bayes regression learning algorithm"
- Naïve Bayes is not well-suited to solving regression problems at all
  - In fact, in the 2-class instance, it learns a linearly-separating hyperplane just like a perceptron
  - That means that for every perceptron, there is an equivalent naïve Bayes classifier (though the proof of this is a bit involved)

36

Issues With Naïve Bayes

- Another big problem with naïve Bayes: often the independence assumption is violated
  - Consider the task of classifying whether or not a certain word is a corporation name
    - e.g. “Google,” “Microsoft,” “IBM,” and “ACME”
  - Two useful features we might want to use are `captialized`, and `all-capitals`
  - Naïve Bayes will assume that these two features are independent of one another, but this clearly isn’t the case (things that are all-caps must also be capitalized)!!

37

Using Conditional Independence

- Clearly the naïve Bayes assumption is too restrictive for many types of problems
- One alternative is to estimate the full-joint probability table
  - But recall that this takes an exponential amount of space
  - Our training probably isn’t large enough to estimate every cell in the FJPT, either! It’s huge!
- Recall, though, absolute vs. conditional independence:
  - Independence: two variables are completely unrelated (NB)
  - Conditional independence: two variables are unrelated to each other, but can be related through a common variable
- Perhaps we can compress the FJPT by using a conditional independence assumption instead?

38

Bayesian Networks

- A Bayesian Belief Network (or Bayes Net) is an AI model that describes the conditional independencies among subsets of variables
- We can use any prior knowledge we might have about the relationships between features
- Yet, we can also take advantage of the inductive learning framework
  - Let the agent learn these probabilities for itself

39

Bayesian Networks

- Bayes Nets are directed acyclic graphs (DAGs):
  - One node for *each* random variable (e.g. feature)
  - A directed edge from cause *A* to its effect *B* represents a direct causal relationships (*B* is conditioned on *A*)
  - Each node is conditionally independent of its non-descendents, given its parents
    - A node is conditioned on its parents, and its descendents are conditioned on it
    - So if there is evidence for its parents, then it is conditionally independent of anything not dependent on it

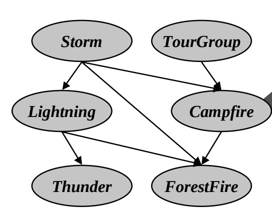
40

Bayesian Networks

- For a set of variables, draw an edge from one variable to one that is conditioned on it
  - Represents direct causal relationships between variables
  - Support for *B* due to the evidence of *A* (e.g. $P(B \mid A)$) is called “diagnostic” or “evidential” support
- Can be used to reason in different ways:
  - Predictive (or causal) reasoning:
    - Forward (top-down) from causes to effects
    - This is what we typically want for inductive learning
  - Can also perform diagnostic reasoning:
    - Backward (bottom-up) from effects to causes

41

Bayesian Networks



	S,T	S,¬T	¬S,T	¬S,¬T
C	0.4	0.1	0.8	0.2

Each node (variable) has a conditional probability table (CPT) over its parents

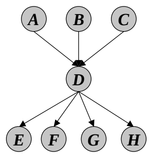
The DAG is a compressed representation of the FJPT over all of the variables, e.g. *P*(*Storm*, *TourGroup*,…, *ForestFire*):

$$P(x_1,...,x_n) = \prod_{i=1}^{n} P(x_i \mid ParentsOf(x_i))$$

42

Bayesian Networks

- To illustrate how compressed Bayes Nets are relative to the FJPT they represent consider the following structure of Boolean random variables:



To represent these 8 variables in a FJPT would take 2^8 entries **= 256**

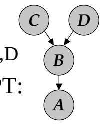
For a Bayes Net, however, the number of values needed for all the CPTs can be computed as:

$$size(BN) = \sum_{v \in BN} 2^{|ParentsOf\,(v)|}$$

$= 2^0 + 2^0 + 2^0 + 2^3 + 2^1 + 2^1 + 2^1 + 2^1 =$ **19**

43

Computing Joint Probabilities



- We want to compute $P(A, B, C, D)$
  - e.g. The joint probability of all events A,…,D
- Recall that the net approximates the FJPT:

$$P(x_1,...,x_n) = \prod_{i=1}^{n} P(x_i \mid ParentsOf\,(x_i))$$

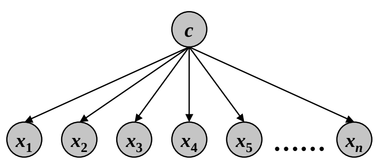
- Then the joint probability reduces to:

$P(A, B, C, D) = P(A \mid B) \times P(B \mid C,D) \times P(C) \times P(D)$

44

Naïve Bayes vs. Bayes Nets

Note that the naïve Bayes classifier is simply a special instance of a Bayes Net: one where the probabilities for the the input features $x_1,…x_n$ (which are all independent) are conditioned on class label c



45

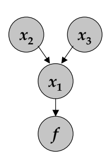
Bayes Net Learning

- Learning probabilities for a Bayesian Network is a straightforward extension of naïve Bayes learning
  - Maintain a probability estimate for each entry in the conditional probability table (CPT) for each node
  - Update the counts in these tables conditioned on the parent nodes of the variable (not the "class label")
  - In the end, divide through to get probabilities
  - We also usually want to initialize counts to a small Laplacian value to "smooth" out zero probabilities

46

Probability Learning Example

- Consider this Bayes Net being trained on the following training set D:



x = TFT $f(x)$ = T x = TFF $f(x)$ = T
x = TTF $f(x)$ = F x = FTF $f(x)$ = F
x = TFT $f(x)$ = T x = FTT $f(x)$ = T
x = TTT $f(x)$ = T x = FTF $f(x)$ = F

$P(x_2)$	0.625

$P(x_3)$	0.5

	x_2, x_3	$\neg x_2, x_3$	$x_2, \neg x_3$	$\neg x_2, \neg x_3$
$P(x_1)$	0.5	1.0	0.33	1.0

	x_1	$\neg x_1$
$P(f)$	0.8	0.33

47

Bayes Net Inference

- In logic, we used inference procedures to show that one sentence α logically followed from others (i.e. $KB \models \alpha$)
- In probability theory, we also need a Bayesian inference procedure to figure probabilities on an effect E, and is an observed cause/causes C

∗ *Since the network is a compressed FJPT, all the information we need is there somewhere…*

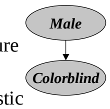
48

Bayes Net Inference

- A nice property about Bayesian networks is that we can calculate the probability of any variable we want!
  - Predictive inference is called top-down because we compute $P(E \mid C)$, the probability of an effect E given causal evidence C (its anscestors)
  - Diagnostic inference is bottom-up because we wish to compute $P(C \mid E)$, the probability of cause C given evidence for an effect E (its descendent)

Inference Example

- Most people who are colorblind are male, so the Bayes net structure to the right is reasonable
- Now we want to perform diagnostic inference to compute the probability of being male given you are colorblind:



$P(M)$	0.45

	M	$\neg M$
$P(C)$	0.1	0.01

$$P(M \mid C) = \frac{P(C \mid M) \times P(M)}{P(C)}$$

$$= \frac{(0.1)(0.45)}{(0.1)(0.45) + (0.01)(0.55)}$$

= 0.89

We don't know $P(C)$ exactly, but we can still compute it by "summing out" over M:

$P(C) = P(C \mid M) \times P(M) + P(C \mid \neg M) \times P(\neg M)$

General Bayesian Inference

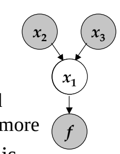
$$P(M \mid C) = \frac{P(C \mid M) \times P(M)}{P(C \mid M) \times P(M) + P(C \mid \neg M) \times P(\neg M)}$$

- The formula above is what we ended up with by using Bayes' Rule and "summing out" over the probability $P(C)$
  - This is equivalent to computing the joint probabilities $P(M,C)$ and $P(\neg M,C)$ and normalizing… we already know how to do that!
- It turns out, we can generalize this to an arbitrary number of variables (not just two)
  - Given evidence $E_1,\ldots,E_n$ and query A, we can compute the joint probabilities $P(A, E_1,\ldots,E_n)$ and $P(\neg A, E_1,\ldots,E_n)$, then normalize… this is identical to $P(A \mid E_1,\ldots,E_n)$

∗ *Note: this only works if all other variables in the Bayesian network are observed!*

Complex Bayesian Inference

- Another unique feature of Bayes nets:
  - If we have missing evidence, we can still perform inference
  - The other variables may have evidence, and might not be conditionally independent anymore



- In general, though, Bayes Net inference is an NP-Hard problem:
  - As more variables go unobserved, the closer we get to calculating the FJPT from the network… and this defeats the purpose!
  - Deterministic and stochastic inference methods that can to this are discussed in CS-731

Bayes Net Inference Examples

- There is a freeware Bayes Net construction and inference system called JavaBayes
  - Clearly implemented in Java
  - **http://www-2.cs.cmu.edu/~javabayes/Home/**
- Comes with many classic "textbook" Bayesian network structures and CPT examples, as well as a few that were trained using ML techniques

Bayes Net Structure Learning

- As with neural networks, the structure of the Bayesian network is very important
  - We need to make sure the directed edges represent dependencies in the real world

∗ *Sometimes we don't know what the real dependencies are… can the agent learn the structure as well??*

Bayes Net Structure Learning

- As with almost all of AI, we can formulate this as an optimization problem:
  - **States:** candidate Bayes Net structures
  - **Actions:** add, reverse, or delete and edge
  - **Objective function:** maximize posterior probability of examples in the training set *D*
- However… as with all optimization problems, we are sensitive to the start state (structure), and run the risk of reaching local optima

55

Summary

- Bayesian probability theory has many applications in modern AI and machine learning
- The basis for the minimum description length principle, which is a theoretical defense of Occam’s razor in learning tasks
- Can also be used in Bayesian classifiers, which compute the most likely classification of examples, if features are treated as “evidence”
  - Continuous features must be descretized
  - Not well-suited to regression problems

56

Summary

- The naïve Bayes classifier is a fast and very popular machine learning algorithm
  - Uses the assumption that all features are independent of one another
- Bayesian Belief Networks (Bayes Nets) are generalizations of naïve Bayes
  - Can learn both probabilities and structures for concepts with rich dependencies
  - Any variable in the network can be queried
  - Not all of the features need to be observed for a probability to be computed

57