Machine Learning

Chapter 06

Bayesian Learning

Bayesian Learning

- Bayesian reasoning provides a probabilistic approach to inference.
- It is based on the assumption that the quantities of interest are governed by probability distributions and that optimal decisions can be made by reasoning about these probabilities together with observed data.
- It is important to machine learning because it provides a quantitative approach to weighing the evidence supporting alternative hypotheses.
- Bayesian reasoning provides the basis for learning algorithms that directly manipulate probabilities, as well as a framework for analyzing the operation of other algorithms that do not explicitly manipulate probabilities.

- Bayesian learning methods are relevant to our study of machine learning for two different reasons.
- First, Bayesian learning algorithms that calculate explicit probabilities for hypotheses, such as the naive Bayes classifier, are among the most practical approaches to certain types of learning problems. For example, Michie et al. (1994) provide a detailed study comparing the naive Bayes classifier to other learning algorithms, including decision tree and neural network algorithms.
- The second reason that Bayesian methods are important to our study of machine learning is that they provide a useful perspective for understanding many learning algorithms that do not explicitly manipulate probabilities.
- For example, in this chapter we analyze algorithms such as the FIND-S and CANDIDATEELIMINATION algorithms to determine conditions under which they output the most probable hypothesis given the training data.

Features of Bayesian learning methods include:

- Each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct. This provides a more flexible approach to learning than algorithms that completely eliminate a hypothesis if it is found to be inconsistent with any single example.
- Prior knowledge can be combined with observed data to determine the final probability of a hypothesis. In Bayesian learning, prior knowledge is provided by asserting
  - a prior probability for each candidate hypothesis, and
  - a probability distribution over observed data for each possible hypothesis. Bayesian methods can accommodate hypotheses that make probabilistic predictions (e.g., hypotheses such as "this pneumonia patient has a 93% chance of complete recovery").
- New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities.
- Even in cases where Bayesian methods prove computationally intractable, they can provide a standard of optimal decision making against which other practical methods can be measured.

Practical Difficulties:

- One practical difficulty in applying Bayesian methods is that they typically require initial knowledge of many probabilities. When these probabilities are not known in advance they are often estimated based on background knowledge, previously available data, and assumptions about the form of the underlying distributions.
- A second practical difficulty is the significant computational cost required to determine the Bayes optimal hypothesis in the general case (linear in the number of candidate hypotheses). In certain specialized situations, this computational cost can be significantly reduced.

BAYES THEOREM

- Bayes theorem provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself.
- To define Bayes theorem
- P(h) to denote the initial probability that hypothesis h holds, before we have observed the training data. P(h) is often called the prior probability of h and may reflect any background knowledge we have about the chance that h is a correct hypothesis. If we have no such prior knowledge, then we might simply assign the same prior probability to each candidate hypothesis.
- P(D) to denote the prior probability that training data D will be observed (i.e., the probability of D given no knowledge about which hypothesis holds).
- P(D/h) to denote the probability of observing data D given some world in which hypothesis h holds.
- The probability P (h/D) that h holds given the observed training data D.
- P (h/D) is called the posterior probability of h, because it reflects our confidence that h holds after we have seen the training data D.
- The posterior probability P(h/D) reflects the influence of the training data D, in contrast to the prior probability P(h) , which is independent of D.
- Bayes theorem is the cornerstone of Bayesian learning methods because it provides a way to calculate the posterior probability P(hlD), from the prior probability P(h), together with P(D) and P(D/h).

Bayes theorem:

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

- P(h ID) increases with P(h) and with P(D/h) according to Bayes theorem.
- It is also reasonable to see that P(h/D) decreases as P(D) increases, because the more probable it is that D will be observed independent of h, the less evidence D provides in support of h.
- In many learning scenarios, the learner considers some set of candidate hypotheses H and is interested in finding the most probable hypothesis h E H given the observed data D (or at least one of the maximally probable if there are several).
- Any such maximally probable hypothesis is called a maximum a posteriori (MAP) hypothesis.
- We can determine the MAP hypotheses by using Bayes theorem to calculate the posterior probability of each candidate hypothesis. More precisely, we will say that h_{MAP} is a MAP hypothesis provided

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

- Notice in the final step above we dropped the term P(D) because it is a constant independent
- In some cases, we will assume that every hypothesis in H is equally probable a priori (P(hi) = P(hj) for all hi and hj in H).
- In this case we can further simplify Equation and need only consider the term P(D/h) to find the most probable hypothesis.
- P(D/h) is often called the likelihood of the data D given h, and any hypothesis that maximizes P(D/h) is called a maximum likelihood (ML) hypothesis,

h_{ML}.

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

An Example

- To illustrate Bayes rule, consider a medical diagnosis problem in which there are two alternative hypotheses:

  (1) that the patient has a particular form of cancer. and

  (2) that the patient does not.
- The available data is from a particular laboratory test with two possible outcomes: + (positive) and - (negative).
- We have prior knowledge that over the entire population of people only .008 have this disease. Furthermore, the lab test is only an imperfect indicator of the disease.
- The test returns a correct positive result in only 98% of the cases in which the disease is actually present and a correct negative result in only 97% of the cases in which the disease is not present. In other cases, the test returns the opposite result.
- The above situation can be summarized by the following probabilities:

$$P(cancer) = .008, \quad P(\neg cancer) = .992$$
$$P(\oplus|cancer) = .98, \quad P(\ominus|cancer) = .02$$
$$P(\oplus|\neg cancer) = .03, \quad P(\ominus|\neg cancer) = .97$$

- Suppose we now observe a new patient for whom the lab test returns a positive result. Should we diagnose the patient as having cancer or not? The maximum a posteriori hypothesis can be found using Equation

$$P(\oplus|cancer)P(cancer) = (.98).008 = .0078$$

$$P(\oplus|\neg cancer)P(\neg cancer) = (.03).992 = .0298$$

Thus, $h_{MAP} = \neg cancer$. The exact posterior probabilities can also be determined by normalizing the above quantities so that they sum to 1 (e.g., $P(cancer|\oplus) = \frac{.0078}{.0078+.0298} = .21$). This step is warranted because Bayes theorem states that the posterior probabilities are just the above quantities divided by the probability of the data, $P(\oplus)$. Although $P(\oplus)$ was not provided directly as part of the problem statement, we can calculate it in this fashion because we know that $P(cancer|\oplus)$ and $P(\neg cancer|\oplus)$ must sum to 1 (i.e., either the patient has cancer or they do not). Notice that while the posterior probability of *cancer* is significantly higher than its prior probability, the most probable hypothesis is still that the patient does not have cancer.

BAYES THEOREM AND CONCEPT LEARNING

- What is the relationship between Bayes theorem and the problem of concept learning? Since Bayes theorem provides a principled way to calculate the posterior probability of each hypothesis given the training data, we can use it as the basis for a straightforward learning algorithm that calculates the probability for each possible hypothesis, then outputs the most probable.

- *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

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

TABLE 6.1
Summary of basic probability formulas.

Brute-Force Bayes Concept Learning

- In particular, assume the learner considers some finite hypothesis space H defined over the instance space X, in which the task is to learn some target concept c : X -> {0,1}.
- As usual, we assume that the learner is given some sequence of training examples ((x1, d1) . . . (xm, dm)) where xi is some instance from X and where di is the target value of xi (i.e., di = c(xi)).
- To simplify the discussion, we assume the sequence of instances (x1 . . . xm) is held fixed, so that the training data D can be written simply as the sequence of target values D = (d1 . . . dm).
- We can design a straightforward concept learning algorithm to output the maximum a posteriori hypothesis, based on Bayes theorem, as follows:

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 the hypothesis h_{MAP} with the highest posterior probability

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

- This algorithm may require significant computation, because it applies Bayes theorem to each hypothesis in H to calculate P(h/D)
- In order specify a learning problem for the BRUTE-FORCE MAP LEARNING algorithm we must specify what values are to be used for P(h) and for P(D/h)
- We may choose the probability distributions P(h) and P(D/h) in any way we wish, to describe our prior knowledge about the learning task.
- Here let us choose them to be consistent with the following assumptions:

  1. The training data D is noise free (i.e., di = c(xi)).

  2. The target concept c is contained in the hypothesis space H

  3. We have no a priori reason to believe that any hypothesis is more probable than any other.
- Given these assumptions, what values should we specify for P(h)? Given no prior knowledge that one hypothesis is more likely than another, it is reasonable to assign the same prior probability to every hypothesis h in H. Furthermore, because we assume the target concept is contained in H we should require that these prior probabilities sum to 1. Together these constraints imply that we should choose

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

- What choice shall we make for P(D/h)? P(D/h) is the probability of observing the target values D = (d1 . . .dm) for the fixed set of instances (x1 . . . xm), given a world in which hypothesis h holds (i.e., given a world in which h is the correct description of the target concept c).
- Since we assume noise-free training data, the probability of observing classification di given h is just 1 if di = h(xi) and 0 if di !=h(xi). Therefore,

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

- In other words, the probability of data D given hypothesis h is 1 if D is consistent with h, and 0 otherwise.

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

First consider the case where h is inconsistent with the training data D. Since Equation (6.4) defines $P(D|h)$ to be 0 when h is inconsistent with D, we have

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0 \text{ if } h \text{ is inconsistent with } D$$

The posterior probability of a hypothesis inconsistent with D is zero.

Now consider the case where h is consistent with D. Since Equation (6.4) defines $P(D|h)$ to be 1 when h is consistent with D, we have

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

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

$$= \frac{1}{|VS_{H,D}|} \text{ if } h \text{ is consistent with } D$$

where $VS_{H,D}$ is the subset of hypotheses from H that are consistent with D (i.e., $VS_{H,D}$ is the version space of H with respect to D as defined in Chapter 2). It is easy to verify that $P(D) = \frac{|VS_{H,D}|}{|H|}$ above, because the sum over all hypotheses of $P(h|D)$ must be one and because the number of hypotheses from H consistent with D is by definition $|VS_{H,D}|$. Alternatively, we can derive $P(D)$ from the theorem of total probability (see Table 6.1) and the fact that the hypotheses are mutually exclusive (i.e., $(\forall i \neq j)(P(h_i \wedge h_j) = 0)$)

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

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

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

- where $|VS_{H,D}|$ is the number of hypotheses from H consistent with D.
- The above analysis implies that under our choice for P(h) and P(D/h), every consistent hypothesis has posterior probability ($1 / V S_{H,D}$), and every inconsistent hypothesis has posterior probability 0.
- Every consistent hypothesis is, therefore, a MAP hypothesis.

MAP Hypotheses and Consistent Learners

- The above analysis shows that in the given setting, every hypothesis consistent with D is a MAP hypothesis.
- This statement translates directly into an interesting statement about a general class of learners that we might call consistent learners.
- We will say that a learning algorithm is a consistent learner provided it outputs a hypothesis that commits zero errors over the training examples.
- Given the above analysis, we can conclude that every consistent learner outputs a MAP hypothesis, if we assume a uniform prior probability distribution over H (i.e., P(hi) = P(hj) for all i, j), and if we assume deterministic, noise free training data (i.e., P(D/h) = 1 if D and h are consistent, and 0 otherwise).

- Consider, for example FIND-S searches the hypothesis space H from specific to general hypotheses, outputting a maximally specific consistent hypothesis (i.e., a maximally specific member of the version space).
- Because FIND-S outputs a consistent hypothesis, we know that it will output a MAP hypothesis under the probability distributions P(h) and P(D/h) defined above.
- Of course FIND-S does not explicitly manipulate probabilities at all -it simply outputs a maximally specific member of the version space.
- However, by identifying distributions for P(h) and P(D/h) under which its output hypotheses will be MAP hypotheses, we have a useful way of characterizing the behavior of FIND-S.

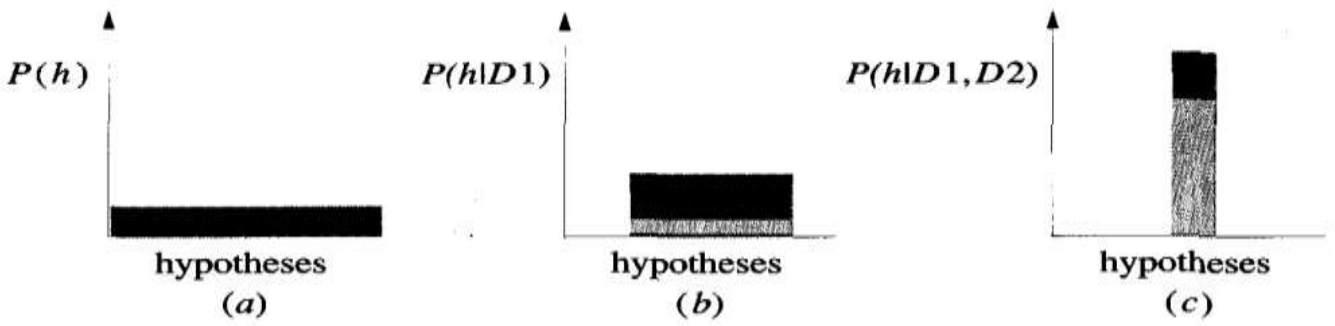


FIGURE 6.1
Evolution of posterior probabilities $P(h|D)$ with increasing training data. (*a*) Uniform priors assign equal probability to each hypothesis. As training data increases first to $D1$ (*b*), then to $D1 \wedge D2$ (*c*), the posterior probability of inconsistent hypotheses becomes zero, while posterior probabilities increase for hypotheses remaining in the version space.

BAYES OPTIMAL CLASSIFIER

- To develop some intuitions consider a hypothesis space containing three hypotheses, h1, h2, and h3.
- Suppose that the posterior probabilities of these hypotheses given the training data are .4, .3, and .3 respectively.
- Thus, h1 is the MAP hypothesis.
- Suppose a new instance x is encountered, which is classified positive by h1, but negative by h2 and h3.
- Taking all hypotheses into account, the probability that x is positive is .4 (the probability associated with hi), and the probability that it is negative is therefore .6.
- The most probable classification (negative) in this case is different from the classification generated by the MAP hypothesis.
- In general, the most probable classification of the new instance is obtained by combining the predictions of all hypotheses, weighted by their posterior probabilities.
- If the possible classification of the new example can take on any value vj from some set V, then the probability P(vj/D) that the correct classification for the new instance is vj, is just

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

- The optimal classification of the new instance is the value vj, for which P (vj/D) is maximum.

Bayes optimal classification:

$$\underset{v_j \in V}{\operatorname{argmax}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

To illustrate in terms of the above example, the set of possible classifications of the new instance is $V = \{\oplus, \ominus\}$, and

$$P(h_1|D) = .4, \quad P(\ominus|h_1) = 0, \quad P(\oplus|h_1) = 1$$
$$P(h_2|D) = .3, \quad P(\ominus|h_2) = 1, \quad P(\oplus|h_2) = 0$$
$$P(h_3|D) = .3, \quad P(\ominus|h_3) = 1, \quad P(\oplus|h_3) = 0$$

therefore

$$\sum_{h_i \in H} P(\oplus|h_i)P(h_i|D) = .4$$

$$\sum_{h_i \in H} P(\ominus|h_i)P(h_i|D) = .6$$

and

$$\underset{v_j \in \{\oplus, \ominus\}}{\operatorname{argmax}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) = \ominus$$

- Any system that classifies new instances according to Equation is called a Bayes optimal classifier, or Bayes optimal learner.
- No other classification method using the same hypothesis space and same prior knowledge can outperform this method on average.
- This method maximizes the probability that the new instance is classified correctly, given the available data, hypothesis space, and prior probabilities over the hypotheses.

NAIVE BAYES CLASSIFIER

- One highly practical Bayesian learning method is the naive Bayes learner, often called the naive Bayes classifier. In some domains its performance has been shown to be comparable to that of neural network and decision tree learning.
- The naive Bayes 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.
- A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values (a1, a2.. .an).
- The learner is asked to predict the target value, or classification, for this new instance.
- The Bayesian approach to classifying the new instance is to assign the most probable target value, VMAP, given the attribute values (al, a2 . . . a,) that describe the instance.

$$v_{MAP} = \underset{v_j \in V}{\text{argmax}} P(v_j | a_1, a_2 \ldots a_n)$$

We can use Bayes theorem to rewrite this expression as

$$v_{MAP} = \underset{v_j \in V}{\text{argmax}} \frac{P(a_1, a_2 \ldots a_n | v_j) P(v_j)}{P(a_1, a_2 \ldots a_n)}$$

$$= \underset{v_j \in V}{\text{argmax}} P(a_1, a_2 \ldots a_n | v_j) P(v_j)$$

- It is easy to estimate each of the P(vj) simply by counting the frequency with which each target value vj occurs in the training data.

The naive Bayes classifier is based on the simplifying assumption that the attribute values are conditionally independent given the target value. In other words, the assumption is that given the target value of the instance, the probability of observing the conjunction $a_1, a_2 \ldots a_n$ is just the product of the probabilities for the individual attributes: $P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$. Substituting this into Equation (6.19), we have the approach used by the naive Bayes classifier.

Naive Bayes classifier:

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

- Where VNB denotes the target value output by the naive Bayes classifier. Notice that in a naive Bayes classifier the number of distinct P(ai/vj) terms that must be estimated from the training data is just the number of distinct attribute values times the number of distinct target values-a much smaller number than if we were to estimate the P(a1, a2 . . . an lvj) terms as first contemplated.
- To summarize, the naive Bayes learning method involves a learning step in which the various P(vj) and P(ai/vj) terms are estimated, based on their frequencies over the training data.
- The set of these estimates corresponds to the learned hypothesis.

An Illustrative Example

- Let us apply the naive Bayes classifier to a concept learning problem we considered during our discussion of decision tree learning: classifying days according to whether someone will play tennis. Table 3.2 from Chapter 3 provides a set of 14 training examples of the target concept PlayTennis, where each day is described by the attributes Outlook, Temperature, Humidity, and Wind. Here we use the naive Bayes classifier and the training data from this table to classify the following novel instance:

$$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$$

Our task is to predict the target value (*yes* or *no*) of the target concept *PlayTennis* for this new instance. Instantiating Equation (6.20) to fit the current task, the target value v_{NB} is given by

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

$$= \underset{v_j \in \{yes, no\}}{\text{argmax}} P(v_j) \quad P(Outlook = sunny|v_j) P(Temperature = cool|v_j)$$

$$P(Humidity = high|v_j) P(Wind = strong|v_j)$$

- Notice in the final expression that ai has been instantiated using the particular attribute values of the new instance. To calculate VNB we now require 10 probabilities that can be estimated from the training data. First, the probabilities of the different target values can easily be estimated based on their frequencies over the 14 training examples

$$P(PlayTennis = yes) = 9/14 = .64$$

$$P(PlayTennis = no) = 5/14 = .36$$

- Similarly, we can estimate the conditional probabilities. For example, those for Wind = strong are

$$P(Wind = strong|PlayTennis = yes) = 3/9 = .33$$

$$P(Wind = strong|PlayTennis = no) = 3/5 = .60$$

Using these probability estimates and similar estimates for the remaining attribute values, we calculate v_{NB} according to Equation (6.21) as follows (now omitting attribute names for brevity)

$$P(yes)\ P(sunny|yes)\ P(cool|yes)\ P(high|yes)\ P(strong|yes) = .0053$$

$$P(no)\ P(sunny|no)\ P(cool|no)\ P(high|no)\ P(strong|no) = .0206$$

Thus, the naive Bayes classifier assigns the target value $PlayTennis = no$ to this new instance, based on the probability estimates learned from the training data. Furthermore, by normalizing the above quantities to sum to one we can calculate the conditional probability that the target value is *no*, given the observed attribute values. For the current example, this probability is $\frac{.0206}{.0206+.0053} = .795$.

ESTIMATING PROBABILITIES

- Up to this point we have estimated probabilities by the fraction of times the event is observed to occur over the total number of opportunities. For example, in the above case we estimated P(Wind = strong/Play Tennis = no) by the fraction nc/n where n = 5 is the total number of training examples for which PlayTennis = no, and nc = 3 is the number of these for which Wind = strong.

m-estimate of probability:

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

- Here nc, and n are defined as before, p is our 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.
- A typical method for choosing p in the absence of other information is to assume uniform priors; that is, if an attribute has k possible values we set p = 1/k.
- For example, in estimating P(Wind = strongl PlayTennis = no) we note the attribute Wind has two possible values, so uniform priors would correspond to choosing p = .5.
- Note that if m is zero, the m-estimate is equivalent to the simple fraction nc/n. If both n and m are nonzero, then the observed fraction nc/n and prior p will be combined according to the weight m.
- The reason m is called the equivalent sample size is that Equation can be interpreted as augmenting the n actual observations by an additional m virtual samples distributed according to p.

BAYESIAN BELIEF NETWORKS

- A Bayesian belief network describes the probability distribution governing a set of variables by specifying a set of conditional independence assumptions along with a set of conditional probabilities.
- In contrast to the naive Bayes classifier, which assumes that all the variables are conditionally independent given the value of the target variable, Bayesian belief networks allow stating conditional independence assumptions that apply to subsets of the variables.
- Thus, Bayesian belief networks provide an intermediate approach that is less constraining than the global assumption of conditional independence made by the naive Bayes classifier, but more tractable than avoiding conditional independence assumptions altogether.
- Bayesian belief networks are an active focus of current research, and a variety of algorithms have been proposed for learning them and for using them for inference.

Conditional Independence

- Let X, Y, and Z be three discrete-valued random variables. We say that X is conditionally independent of Y given Z if the probability distribution governing X is independent of the value of Y given a value for Z; that is, if

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

where $x_i \in V(X)$, $y_j \in V(Y)$, and $z_k \in V(Z)$. We commonly write the above expression in abbreviated form as $P(X|Y, Z) = P(X|Z)$. This definition of conditional independence can be extended to sets of variables as well. We say that the set of variables $X_1 \ldots X_l$ is conditionally independent of the set of variables $Y_1 \ldots Y_m$ given the set of variables $Z_1 \ldots Z_n$ if

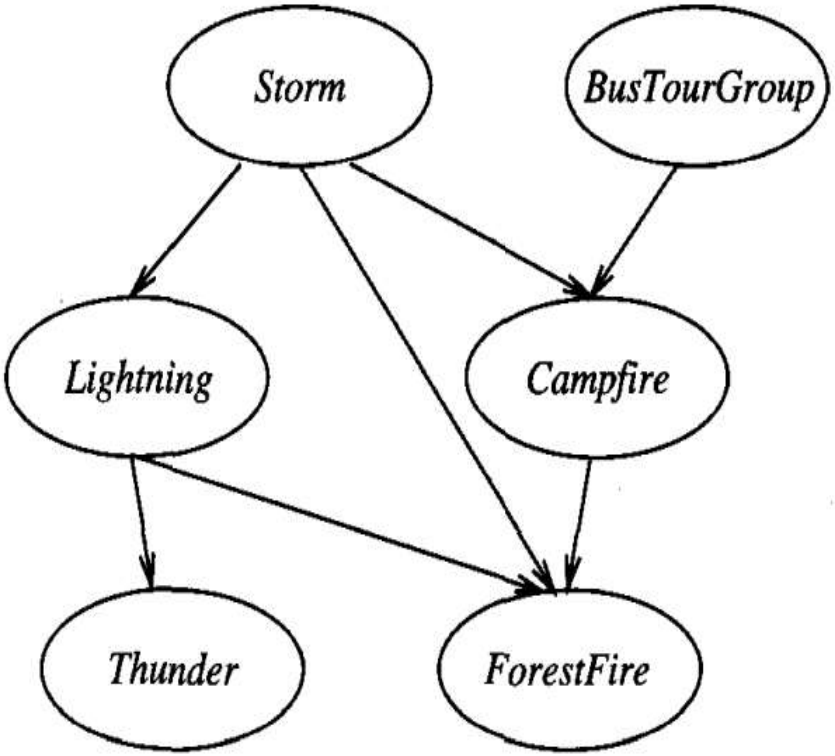
$$P(X_1 \ldots X_l | Y_1 \ldots Y_m, Z_1 \ldots Z_n) = P(X_1 \ldots X_l | Z_1 \ldots Z_n)$$

- Note the correspondence between this definition and our use of conditional , independence in the definition of the naive Bayes classifier. The naive Bayes classifier assumes that the instance attribute A1 is conditionally independent of instance attribute A2 given the target value V. This allows the naive Bayes classifier to calculate P(A1, A2/V)

$$P(A_1, A_2|V) = P(A_1|A_2, V)P(A_2|V) \quad (6.23)$$

$$= P(A_1|V)P(A_2|V) \quad (6.24)$$

Equation (6.23) is just the general form of the product rule of probability from Table 6.1. Equation (6.24) follows because if A_1 is conditionally independent of A_2 given V, then by our definition of conditional independence $P(A_1|A_2, V) = P(A_1|V)$.



	S,B	$S,\neg B$	$\neg S,B$	$\neg S,\neg B$
C	0.4	0.1	0.8	0.2
$\neg C$	0.6	0.9	0.2	0.8

FIGURE 6.3

A Bayesian belief network. The network on the left represents a set of conditional independence assumptions. In particular, each node is asserted to be conditionally independent of its nondescendants, given its immediate parents. Associated with each node is a conditional probability table, which specifies the conditional distribution for the variable given its immediate parents in the graph. The conditional probability table for the *Campfire* node is shown at the right, where *Campfire* is abbreviated to C, *Storm* abbreviated to S, and *BusTourGroup* abbreviated to B.

Representation

- A Bayesian belief network (Bayesian network for short) represents the joint probability distribution for a set of variables.
- For example, the Bayesian network in Figure 6.3 represents the joint probability distribution over the boolean variables Storm, Lightning, Thunder, ForestFire, Campfire, and BusTourGroup. In general, a Bayesian network represents the joint probability distribution by specifying a set of conditional independence assumptions (represented by a directed acyclic graph), together with sets of local conditional probabilities.
- Each variable in the joint space is represented by a node in the Bayesian network.
- For each variable two types of information are specified.
- First, the network arcs represent the assertion that the variable is conditionally independent of its non descendants in the network given its immediate predecessors in the network.
- We say Xi is a descendant of Y if there is a directed path from Y to X.
- Second, a conditional probability table is given for each variable, describing the probability distribution for that variable given the values of its immediate predecessors.
- The joint probability for any desired assignment of values (y1, . . . , yn) to the tuple of network variables (Y1 . . . Yn) can be computed by the formula

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

- where Parents(Yi) denotes the set of immediate predecessors of Yi in the network. Note the values of P(yi/ Parents(Yi)) are precisely the values stored in the conditional probability table associated with node Yi.
- To illustrate, the Bayesian network in Figure 6.3 represents the joint probability distribution over the boolean variables Storm, Lightning, Thunder, Forest, Fire, Campfire, and BusTourGroup. Consider the node Campfire. The network nodes and arcs represent the assertion that Campfire is conditionally independent of its non descendants Lightning and Thunder, given its immediate parents Storm and BusTourGroup. This means that once we know the value of the variables Storm and BusTourGroup, the variables Lightning and Thunder provide no additional information about Campfire. The right side of the figure shows the conditional probability table associated with the variable Campfire. The top left entry in this table, for example, expresses the assertion that

$$P(Campfire = True | Storm = True, BusTourGroup = True) = 0.4$$

- Note this table provides only the conditional probabilities of Campfire given its parent variables Storm and BusTourGroup.

THE EM ALGORITHM

- In many practical learning settings, only a subset of the relevant instance features might be observable.
- For example, in training or using the Bayesian belief network of Figure 6.3, we might have data where only a subset of the network variables Storm, Lightning, Thunder, ForestFire, Campfire, and BusTourGroup have been observed. Many approaches have been proposed to handle the problem of learning in the presence of unobserved variables.
- the EM algorithm 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.
- The EM algorithm is also the basis for many unsupervised clustering algorithms and it is the basis for the widely used Baum-Welch forward-backward algorithm for learning Partially Observable Markov Models.

Estimating Means of k Gaussians

- The easiest way to introduce the EM algorithm is via an example. Consider a problem in which the data D is a set of instances generated by a probability distribution that is a mixture of k distinct Normal distributions.
- This problem setting is illustrated in Figure 6.4 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 xi is generated according to this selected distribution. This process is repeated to generate a set of data points as shown in the figure
- 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 σ2, and where σ2 is known.
- The learning task is to output a hypothesis h = (μ1, . . . μk) 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).

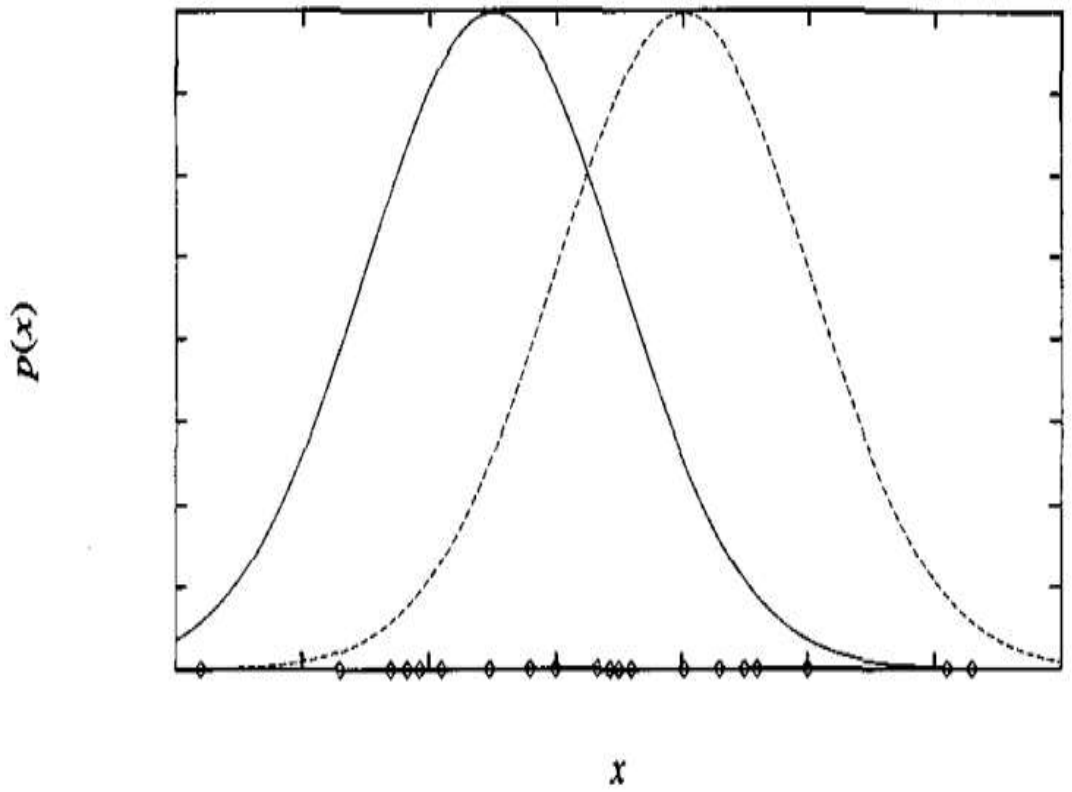


FIGURE 6.4

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.

Note it is easy to calculate the maximum likelihood hypothesis for the mean of a single Normal distribution given the observed data instances $x_1, x_2, \ldots, x_m$ drawn from this single distribution. This problem of finding the mean of a single distribution is just a special case of the problem discussed in Section 6.4, Equation (6.6), where we showed that the maximum likelihood hypothesis is the one that minimizes the sum of squared errors over the m training instances. Restating Equation (6.6) using our current notation, we have

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

In this case, the sum of squared errors is minimized by the sample mean

$$\mu_{ML} = \frac{1}{m} \sum_{i=1}^{m} x_i$$

Applied to the problem of estimating the two means for Figure 6.4, the EM algorithm first initializes the hypothesis to $h = \langle \mu_1, \mu_2 \rangle$, where μ_1 and μ_2 are arbitrary initial values. It then iteratively re-estimates h by repeating the following two steps until the procedure converges to a stationary value for h.

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.

Let us examine how both of these steps can be implemented in practice. Step 1 must calculate the expected value of each z_{ij}. This $E[z_{ij}]$ is just the probability that instance x_i was generated by the jth Normal distribution

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

$$= \frac{e^{-\frac{1}{2\sigma^2}(x_i - \mu_j)^2}}{\sum_{n=1}^{2} e^{-\frac{1}{2\sigma^2}(x_i - \mu_n)^2}}$$

Thus the first step is implemented by substituting the current values $\langle \mu_1, \mu_2 \rangle$ and the observed x_i into the above expression.

In the second step we use the $E[z_{ij}]$ calculated during Step 1 to derive a new maximum likelihood hypothesis $h' = \langle \mu'_1, \mu'_2 \rangle$. As we will discuss later, the maximum likelihood hypothesis in this case is given by

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

General Statement of EM Algorithm

- The EM algorithm can be applied in many settings where we wish to estimate some set of parameters Θ that describe an underlying probability distribution, given only the observed portion of the full data produced by this distribution.
- In the above two-means example the parameters of interest were Θ = (μ1, μ2), and the full data were the triples (xi, zi1, zi2) of which only the xi were observed. In general let X = {x1, . . . , xm} denote the observed data in a set of m independently drawn instances, let Z = {z1, . . . , zm} denote the unobserved data in these same instances, and let Y = X U Z denote the full data.
- Note the unobserved Z can be treated as a random variable whose probability distribution depends on the unknown parameters Θ and on the observed data X. Similarly, Y is a random variable because it is defined in terms of the random variable Z.
- In the remainder of this section we describe the general form of the EM algorithm. We use h to denote the current hypothesized values of the parameters Θ, and h' to denote the revised hypothesis that is estimated on each iteration of the EM algorithm.

- The EM algorithm searches for the maximum likelihood hypothesis h' by seeking the h' that maximizes E[ln P(Y/h')]. This expected value is taken over the probability distribution governing Y, which is determined by the unknown parameters Θ.
- What is the probability distribution governing Y? In general we will not know this distribution because it is determined by the parameters Θ that we are trying to estimate. Therefore, the EM algorithm uses its current hypothesis h in place of the actual parameters Θ to estimate the distribution governing Y. Let us define a function Q(h'/h) that gives E[ln P(Y/h')] as a function of h', under the assumption that Θ= h and given the observed portion X of the full data Y.

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

- We write this function Q in the form Q(h'/h) to indicate that it is defined in part by the assumption that the current hypothesis h is equal to Θ. In its general form, the EM algorithm repeats the following two steps until convergence:

Step 1: *Estimation (E) step:* Calculate $Q(h'|h)$ using the current hypothesis h and the observed data X to estimate the probability distribution over Y.

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

Step 2: *Maximization (M) step:* Replace hypothesis h by the hypothesis h' that maximizes this Q function.

$$h \leftarrow \underset{h'}{\operatorname{argmax}}\, Q(h'|h)$$

When the function Q is continuous, the EM algorithm converges to a stationary point of the likelihood function $P(Y|h')$. When this likelihood function has a single maximum, EM will converge to this global maximum likelihood estimate for h'. Otherwise, it is guaranteed only to converge to a local maximum. In this respect, EM shares some of the same limitations as other optimization methods such as gradient descent, line search, and conjugate gradient discussed in Chapter 4.