MODULE 4: BAYESIAN LEARNING

References
1. Tom M. Mitchell, Machine Learning, India Edition 2013, McGraw Hill Education.

4.1 INTRODUCTION

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 CANDIDATE ELIMINATION Algorithms of Chapter 2 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 (1) a prior probability for each candidate hypothesis, and (2) 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.

4.2 BAYES THEOREM

In machine learning we are often interested in determining the best hypothesis from some space H, given the observed training data D. One way to specify what we mean by the ***best*** hypothesis is to say that we demand the ***most probable*** hypothesis, given the data ***D*** plus any initial knowledge about the prior probabilities of the various hypotheses in H. Bayes theorem provides a direct method for calculating such probabilities. More precisely, 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 precisely, let us first introduce a little notation. We shall write ***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. Similarly, we will write ***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). Next, we will write ***P(D|h)*** to denote the probability of observing data ***D*** given some world in which hypothesis ***h*** holds. More generally, we write ***P(x|y)*** to denote the probability of x given ***y.*** In machine learning problems we are interested in 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.*** Notice 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(h|D),*** from the prior probability ***P(h),*** together with ***P(D)*** and ***P(D|h).***

Bayes theorem:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \qquad \text{------------- (1)}$$

As one might intuitively expect, ***P(h|D)*** 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€*** 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 aposteriori*** (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

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

--------------------- (2)

Notice in the final step above we dropped the term ***P (D)*** because it is a constant independent of ***h.*** In some cases, we will assume that every hypothesis in H is equally probable a priori ***(P(h_i)*** = ***P(h_j)*** for all h_i and h_j in H). In this case we can further simplify Equation (2) 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}.

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

--------------------------- (3)

4.2.1 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: $\oplus$ (positive) and $\ominus$ (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:

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

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 aposteriori hypothesis can be found using Equation (2):

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

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

Thus, h_{MAP} = ¬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($\oplus$).

4.3 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.

4.3.1 Brute-Force Bayes Concept Learning

Consider the concept learning problem first introduced in Chapter 2. 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 $((x_1 d_1) \ldots (x_m, d_m))$ where x_i is some instance from X and where d_i is the target value of x_i (i.e., $d_i = c(x_i)$). To simplify the discussion in this section, we assume the sequence of instances $(x_l \ldots x_m)$ is held fixed, so that the training data D can be written simply as the sequence of target values D = $(d_l \ldots d_m)$. It can be shown that this simplification does not alter the main conclusions of this section. 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)*** **.** While this may prove impractical for large hypothesis spaces, the algorithm is still of interest because it provides a standard against which we may judge the performance of other concept learning algorithms.

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)*** (as we shall see, ***P(D)*** will be determined once we choose the other two). 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., $d_i = c(x_i)$).
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 = $(d_1 \ldots d_m)$ for the fixed set of instances $(x_1 \ldots x_m)$,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 d_i given ***h*** is just ***1*** if $d_i = h(x_i)$ and 0 if $d_i \neq h(x_i)$. Therefore,

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

------------------------ (4)

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

Given these choices for ***P(h)*** and for ***P(D|h)*** we now have a fully-defined problem for the above **BRUTE-FORCE MAP LEARNING** algorithm. Let us consider the first step of this algorithm, which uses Bayes theorem to compute the posterior probability ***P(h|D)*** of each hypothesis ***h*** given the observed training data ***D.***

Recalling Bayes theorem, we have

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

First consider the case where *h* is inconsistent with the training data *D.* Since Equation (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 *(4)* defines *P(D|h)* to be 1 when *h* is consistent with *D,* we have

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

where $VS_{H,D}$ is the subset of hypotheses from H that are consistent with *D* (i.e., $VS_{H,D}$ 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 $|VS_{H,D}|$.

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

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

$$P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$ ---------------- (5)

4.3.2 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).*

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.

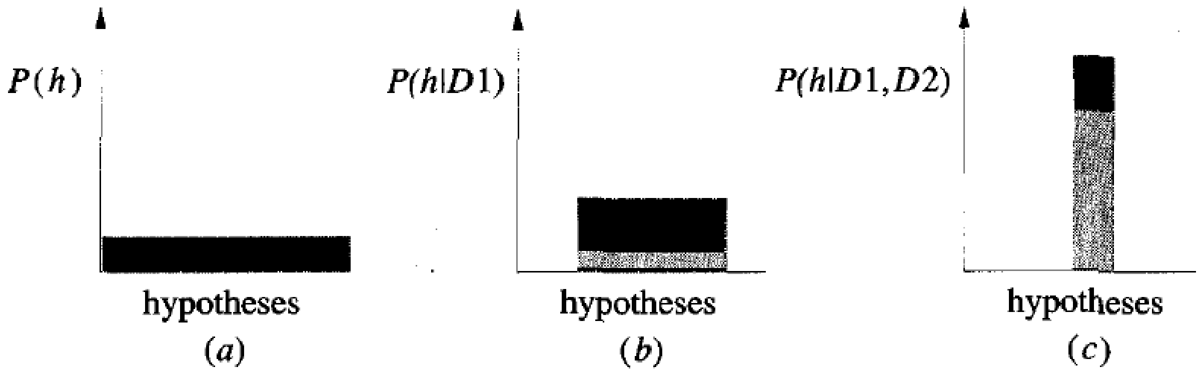


Figure 4.1: Evolution of posterior probabilities *P(h|D)* with increasing training data. *(a)* Uniform priors assign equal probability to each hypothesis. As training data increases first to *Dl* (b), then to *D1^D2* (c), the posterior probability of inconsistent hypotheses becomes zero, while posterior probabilities increase for hypotheses remaining in the version space.

4.4 MAXIMUM LIKELIHOOD AND LEAST-SQUARED ERROR HYPOTHESES

A straight forward Bayesian analysis will show that under certain assumptions any learning algorithm that minimizes the squared error between the output hypothesis predictions and the training data will output a maximum likelihood hypothesis. Consider any real-valued target function *f*. Training examples <*xi, di*>, where *di* is noisy training value *di*= *f*(*xi*) + *ei* where *ei* is random variable (noise) drawn independently for each *xi* according to some Gaussian distribution with mean=0.Then the maximum likelihood hypothesis h_{ML} is the one that minimizes the sum of squared errors:

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

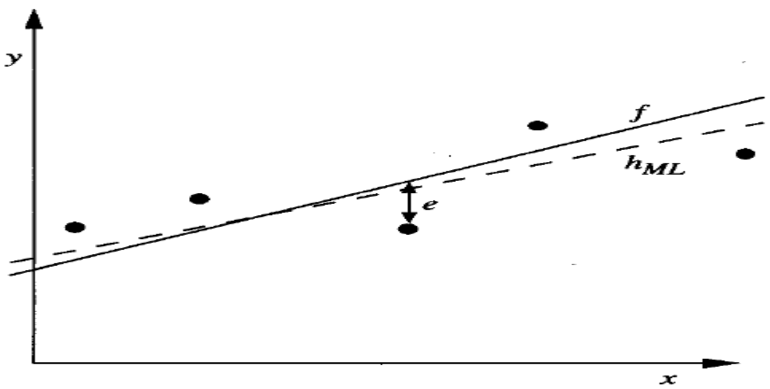


Figure 4.2: Learning a real-valued function.

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

Maximize natural log of this instead

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

-------------- (6)

4.5 MAXIMUM LIKELIHOOD HYPOTHESES FOR PREDICTING PROBABILITIES

Consider predicting survival probability from patient data

•Training examples <*xi*, *di*>, where *di* is 1 or 0

•Want to train neural network to output a *probability* given *xi* (not a 0 or 1)

Recall that in the maximum likelihood, least-squared error analysis of the previous section, we made the simplifying assumption that the instances ($x_1 \ldots x_m$) were fixed. This enabled us to characterize the data by considering only the target values di. Although we could make a similar simplifying assumption in this case, let us avoid it here in order to demonstrate that it has no impact on the final outcome. Thus treating both xi and di as random variables, and assuming that each training example is drawn independently, we can write P(D|h) as

$$
P(D|h) = \prod_{i=1}^{m} P(x_i, d_i|h)
$$

------------------------- (7)

It is reasonable to assume, furthermore, that the probability of encountering any particular instance xi is independent of the hypothesis h. For example, the probability that our training set contains a particular ***patient*** xi is independent of our hypothesis about survival rates (though of course the ***survival d,*** of the patient does depend strongly on ***h).*** When ***x*** is independent of ***h*** we can rewrite the above expression as

$$P(D|h) = \prod_{i=1}^{m} P(x_i, d_i|h) = \prod_{i=1}^{m} P(d_i|h, x_i)P(x_i)$$

------------------------- (8)

Now what is the probability ***P(di|h, xi)*** of observing ***di*** = ***1*** for a single instance ***xi,*** given a world in which hypothesis ***h*** holds? Recall that ***h*** is our hypothesis regarding the target function, which computes this very probability. Therefore, ***P(di*** = ***1| h, xi)*** = ***h(xi),*** and in general

$$P(d_i|h, x_i) = \begin{cases} h(x_i) & \text{if } d_i = 1 \\ (1 - h(x_i)) & \text{if } d_i = 0 \end{cases}$$

---------------------- (9)

In order to substitute this into the Equation ***(6.8)*** for ***P(Dlh),*** let us first
" re-express it in a more mathematically manipulable form, as

In this case can show

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} d_i \ln h(x_i) + (1 - d_i) \ln(1 - h(x_i))$$

Weight update rule for a sigmoid unit:

$$\boxed{w_{jk} \leftarrow w_{jk} + \Delta w_{jk}}$$

where

$$\Delta w_{jk} = \eta \sum_{i=1}^{m} (d_i - h(x_i))\ x_{ijk}$$

4.6 MINIMUM DESCRIPTION LENGTH PRINCIPLE

Recall from Chapter 3 the discussion of Occam's razor, a popular inductive bias that can be summarized as "choose the shortest explanation for the observed data." In that chapter we discussed several arguments in the long-standing debate regarding Occam's razor. Here we consider a Bayesian perspective on this issue and a closely related principle called the Minimum Description Length (MDL) principle. The Minimum Description Length principle is motivated by interpreting the definition of h_{MAP} in the light of basic concepts from information theory. Consider again the now familiar definition of h_{MAP}.

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

which can be equivalently expressed in terms of maximizing the $\log_2$

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

or alternatively, minimizing the negative of this quantity

$$h_{MAP} = \underset{h \in H}{\operatorname{argmin}} \; -\log_2 P(D|h) - \log_2 P(h)$$
----------- (10)

Somewhat surprisingly, Equation (10) can be interpreted as a statement that short hypotheses are preferred, assuming a particular representation scheme for encoding hypotheses and data. To explain this, let us introduce a basic result from information theory: Consider the problem of designing a code to transmit messages drawn at random, where the probability of encountering message ***i*** is ***pi.*** We are interested here in the most compact code; that is, we are interested in the code that minimizes the expected number of bits we must transmit in order to encode a message drawn at random. Clearly, to minimize the expected code length we should assign shorter codes to messages that are more probable.

The Minimum Description Length (MDL) principle recommends choosing the hypothesis that minimizes the sum of these two description lengths. Of course to apply this principle in practice we must choose specific encodings or representations appropriate for the given learning task. Assuming we use the codes C_1 and C_2 to represent the hypothesis and the data given the hypothesis, we can state the MDL principle as

Minimum Description Length principle: Choose h_{MDL} where

$$h_{MDL} = \underset{h \in H}{\operatorname{argmin}} \; L_{C_1}(h) + L_{C_2}(D|h)$$
------> (11)

The above analysis shows that if we choose C_1 to be the optimal encoding of hypotheses C_H, and if we choose C_2 to be the optimal encoding $C_{D|h}$, then $h_{MDL} = h_{MAP}$.

4.7 NAIVE BAYES CLASSIFIER

- Highly bayesian learning method is the naïve Bayes learner often called the naïve Bayes Classifier.
- The naïve 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 <a_1, a_2 , a_3 ,------- a_n >. 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, V_{MAP}, given the attribute values <a_1, a_2 , a_3 ,------- a_n > that describe the instance.

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

- We can use Bayes theorem to rewrite this expression as

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

Naive Bayes classifier:

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

4.7.1 An Illustrative Example

Let us apply the naive Bayes classifier to a concept learning problem i.e., classifying days according to whether someone will play tennis. The below table 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

Table 1: Play Tennis Dataset

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
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	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	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

Use the naive Bayes classifier and the training data from this table to classify the following novel instance:

< Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong >

Our task is to predict the target value ***(yes or no)*** of the target concept ***PlayTennis*** for this new instance

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

$$V_{NB} = \underset{v_j \in \{yes, no\}}{\operatorname{argmax}} P(v_j)\, P(Outlook=sunny|v_j)\, P(Temperature=cool|v_j)\, P(Humidity=high|v_j)\, P(Wind=strong|v_j)$$

The probabilities of the different target values can easily be estimated based on their frequencies over the 14 training examples

- P(P1ayTennis = yes) = 9/14 = 0.64
- P(P1ayTennis = no) = 5/14 = 0.36

Similarly, estimate the conditional probabilities. For example, those for Wind = strong

- P(Wind = strong | PlayTennis = yes) = 3/9 = 0.33
- P(Wind = strong | PlayTennis = no) = 3/5 = 0.60

Calculate VNB according to Equation (1)

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

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

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.By normalizing the above quantities to sum to one, calculate the conditional probability that the target value is ***no***, given the observed attribute values

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

Estimating Probabilities

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. When nc = 0, then nc /n will be zero and this probability term will dominate the quantity calculated in Equation (2) requires multiplying all the other probability terms by this zero value. To avoid this difficulty we can adopt a Bayesian approach to estimating the probability, using the ***m-estimate*** defined as follows

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

4.8 BAYESIAN BELIEF NETWORKS

- The naive Bayes classifier makes significant use of the assumption that the values of the attributes $a_1 \ldots a_n$ are conditionally independent given the target value v.
- This assumption dramatically reduces the complexity of learning the target function.
- 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.
- Bayesian belief networks allow stating conditional independence assumptions that apply to subsets of the variables.

Notation

Consider an arbitrary set of random variables $Y_1 \ldots Yn$, where each variable Yi can take on the set of possible values V(Yi). The joint space of the set of variables Y to be the cross product $V(Y_1)$ x $V(Y_2)$ x. . . V(Yn). In other words, each item in the joint space corresponds to one of the possible assignments of values to the tuple of variables $(Y_1 \ldots Yn)$. The probability distribution over this joint' space is called the joint probability distribution. The joint probability distribution specifies the probability for each of the possible variable bindings for the tuple $(Y_1 \ldots Yn)$. A Bayesian belief network describes the joint probability distribution for a set of variables.

4.8.1 Conditional Independence

Let X, Y, and Z be three discrete-valued random variables. 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,

$$x_i \in V(X),\ y_j \in V(Y),\ \text{and}\ z_k \in V(Z).$$

The above expression is written in abbreviated form as

$$P(X \mid Y, Z) = P(X \mid Z)$$

Conditional independence can be extended to sets of variables. The set of variables $X_1 \ldots X_l$ is conditionally independent of the set of variables $Y_1 \ldots Ym$ given the set of variables $Z1 \ldots Zn$ if

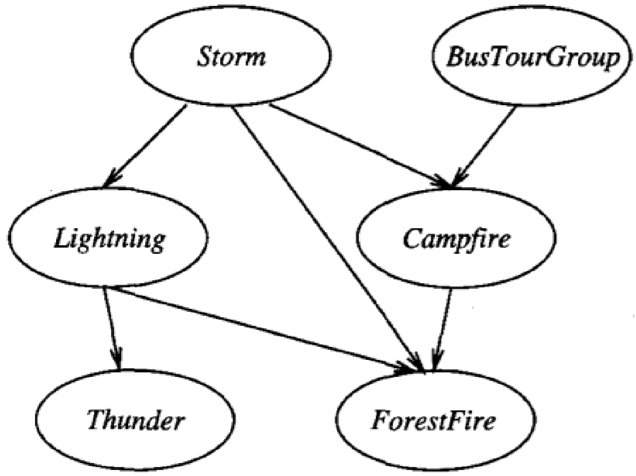
$$P(X_1 \ldots X_l | Y_1 \ldots Y_m, Z_1 \ldots Z_n) = P(X_1 \ldots X_l | Z_1 \ldots Z_n)$$

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(Al, A2 | V) as follows,

$$\begin{aligned} P(A_1, A_2|V) &= P(A_1|A_2, V)P(A_2|V) \\ &= P(A_1|V)P(A_2|V) \end{aligned}$$

Representation

A Bayesian belief network represents the joint probability distribution for a set of variables. 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

Figure 4.3: Bayesian Belief Network

The Bayesian network in above figure represents the joint probability distribution over the boolean variables *Storm, Lightning, Thunder, ForestFire, Campfire,* and *BusTourGroup*. A Bayesian network (BN) represents the joint probability distribution by specifying a set of *conditional independence assumptions*.

BN 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.

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. A ***conditional probability table*** **(CPT)** is given for each variable, describing the probability distribution for that variable given the values of its immediate predecessors.

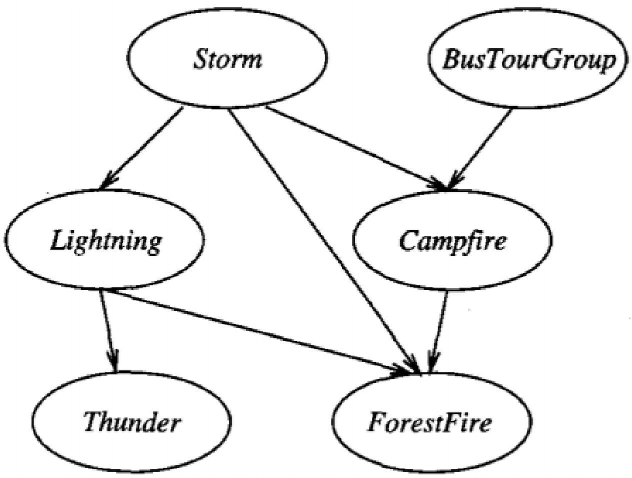
The joint probability for any desired assignment of values (y_1, . . . , yn) to the tuple of network variables (Y_1 . . . Ym) can be computed by the formula

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

Where, Parents(Yi) denotes the set of immediate predecessors of Yi in the network.

Example:

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***.



	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

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 conditional probability table associated with the variable ***Campfire.*** The assertion is

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

Inference

- Use a Bayesian network to infer the value of some target variable (e.g., ForestFire) given the observed values of the other variables. Inference can be straightforward if values for all of the other variables in the network are known exactly.
- A Bayesian network can be used to compute the probability distribution for any subset of network variables given the values or distributions for any subset of the remaining variables. An arbitrary Bayesian network is known to be NP-hard.

4.9 THE EM ALGORITHM

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.

4.9.1 Estimating Means of k Gaussians

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 4.4 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 below figure.

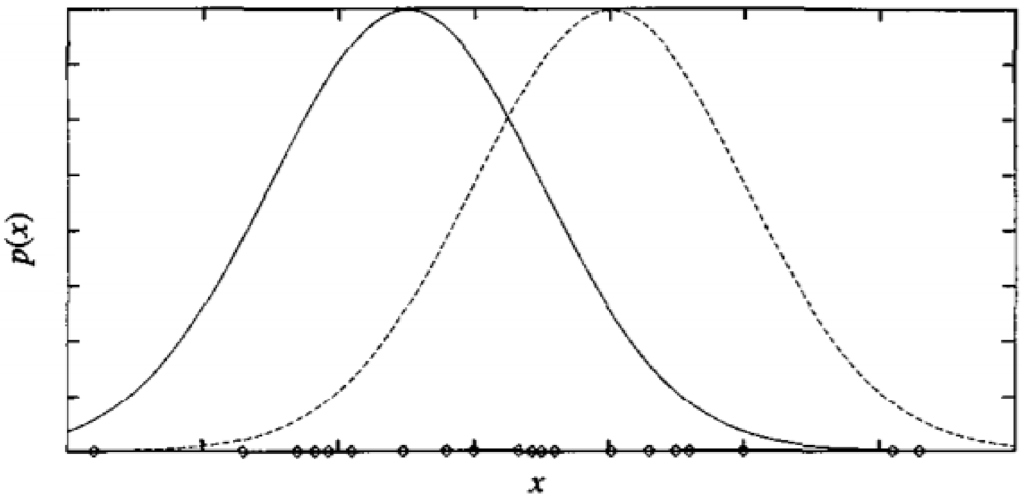


Figure 4.4: Probability Distribution of N data points

To simplify, consider the special case

- The selection of the single Normal distribution at each step is based on choosing each with uniform probability
- 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).

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

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

$$\mu_{ML} = \frac{1}{m} \sum_{i=1}^{m} x_i \qquad (2)$$

4.9.2 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.

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

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

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 $h' = \langle \mu'_1, \mu'_2 \rangle$.
maximum likelihood hypothesis in this case is given by

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

Acknowledgement

The diagrams and tables are taken from the textbooks specified in the references section.

Prepared by:

Rakshith M D

Department of CS&E

SDMIT, Ujire