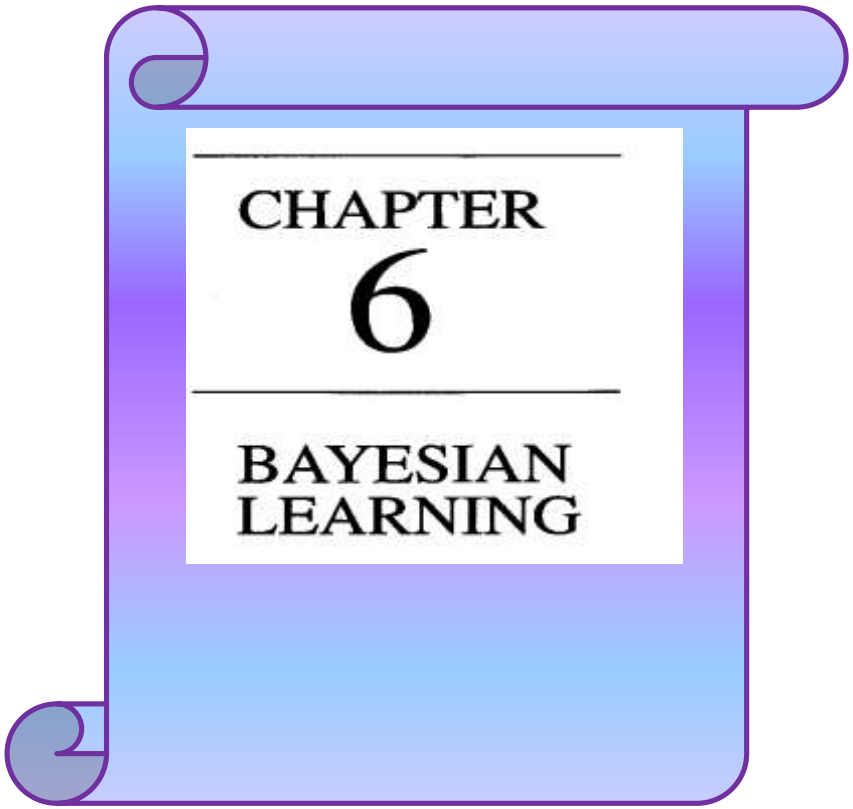
Bayesian Learning n>=30

U
N
I
T

3



B. Tech III Year
CS 601 PC

UNIT – III Bayesian learning

- Introduction,----------3
- Bayes theorem, ----------6
- Bayes theorem and concept learning, ---------12
- Maximum Likelihood and least squared error hypotheses,
- Maximum likelihood hypotheses for predicting probabilities, -----
- **minimum description length principle**,
- Bayes optimal classifier,
- Gibbs algorithm,
- **Naïve Bayes classifier**,
  - an example: learning to classify text,
- **Bayesian belief networks**,
- the **EM algorithm**

6.1 Bayesian Learning – Introduction n>=30

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 ML 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 methods is important to understand and characterize the operation of many algorithms in ML.

Features of Bayesian learning methods:

- 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. < specific , general , specific ,,,,,,,,,,,?,,,,?>

- Prior knowledge can be combined with observed data to determine the final probability of 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.

Practical Difficulties :

1) in applying Bayesian methods is that they typically require initial knowledge of many probabilities.

2) the significant computational cost required to determine the Bayes optimal hypothesis in the general case .
(linear in the number of candidate hypotheses)

6.2 BAYES THEOREM

Bayes theorem is the cornerstone of Bayesian learning methods.

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:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \qquad (6.1)$$

As one might intuitively expect, ***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 ϵ 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 ***MAP is a MAP hypothesis provided***

$$
\begin{aligned}
h_{MAP} &\equiv \underset{h \in H}{\operatorname{argmax}} P(h|D) \\
&= \underset{h \in H}{\operatorname{argmax}} \frac{P(D|h)\,P(h)}{P(D)} \\
&= \underset{h \in H}{\operatorname{argmax}} P(D|h)\,P(h) \qquad (6.2)
\end{aligned}
$$

6.2.1 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:

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

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 (6.2):

$$P(\oplus|cancer)P(cancer) = (.98).008 = .0078$$
$$P(\oplus|\neg cancer)P(\neg cancer) = (.03).992 = .0298$$

$$h_{MAP} = \neg cancer.$$

The exact posterior probabilities can also be determined by normalizing the above quantities so that they sum to 1

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

2)Bayes theorem states that the posterior probabilities are just the above quantities divided by the probability of the data,

$$P(cancer|\oplus) \text{ and } P(\neg cancer|\oplus) \text{ must sum to 1}$$

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.**

- *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

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

6.3 BAYES THEOREM AND CONCEPT LEARNING

Is There any Relationship between Bayes theorem and 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** it has be most general

a brute-force Bayesian concept learning algorithm,

6.3.1 Brute-Force Bayes Concept Learning

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

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

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)*** *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.

Can we choose them to be **consistent** by making assumptions???

Y

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

Prior probablility

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

What choice shall we make for $P(D|h)$? $P(D|h)$ is the probability of observing the target values $D = \langle d_1 \ldots d_m \rangle$ for the fixed set of instances $\langle x_1 \ldots x_m \rangle$, 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,

likelihood

$$P(D|h) = \begin{cases} 1 \text{ if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 \text{ otherwise} \end{cases} \tag{6.4}$$

Classification is inferring the Boolean valued function

In other words, the probability of data D given hypothesis h is 1 if D is consistent

Recalling Bayes theorem, we have

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

Conditional probablility

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

CASE 1

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

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

CASE 2

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

Likelihood =1

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

Subset of hypothesis

Conditional probablility

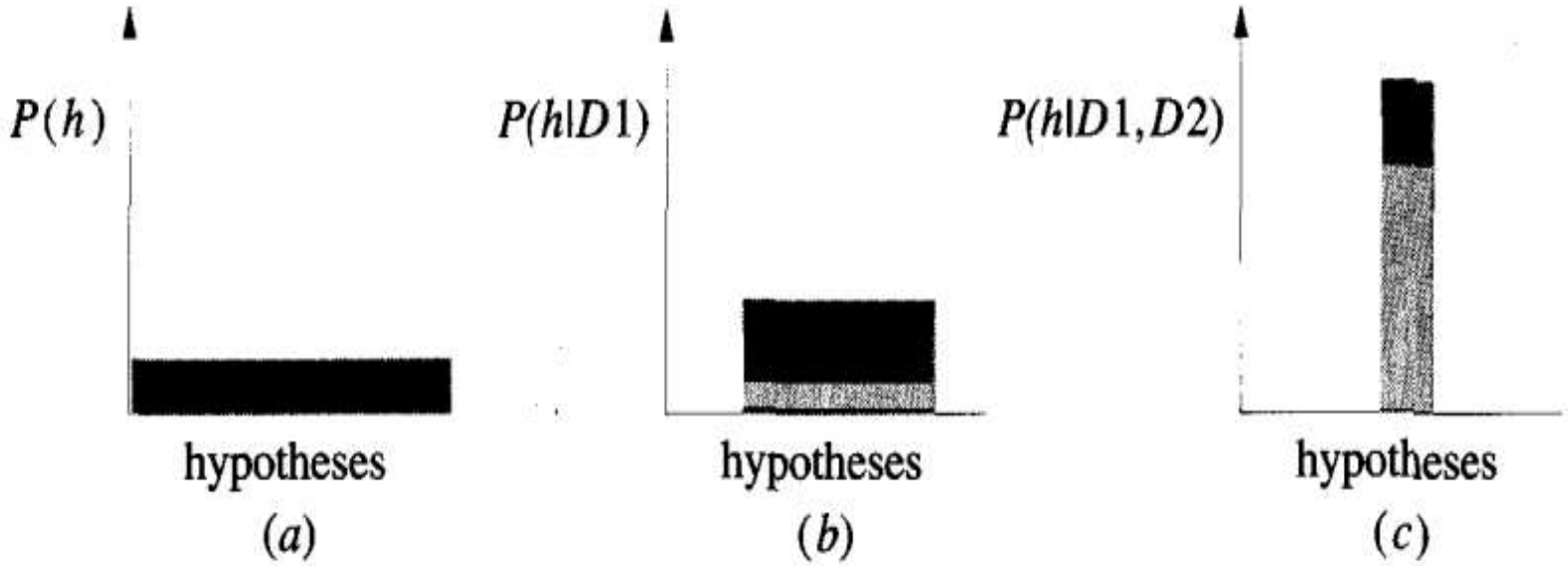
$$= \frac{1}{|VS_{H,D}|} \text{ if } h \text{ is consistent with } D$$

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

$$
P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases} \tag{6.5}
$$



Evolution of posterior probabilities P(hlD) with increasing training data.

(a)Uniform priors assign equal probability to each hypothesis.

(b)As training data increases first to Dl (b), then to Dl ^ D2

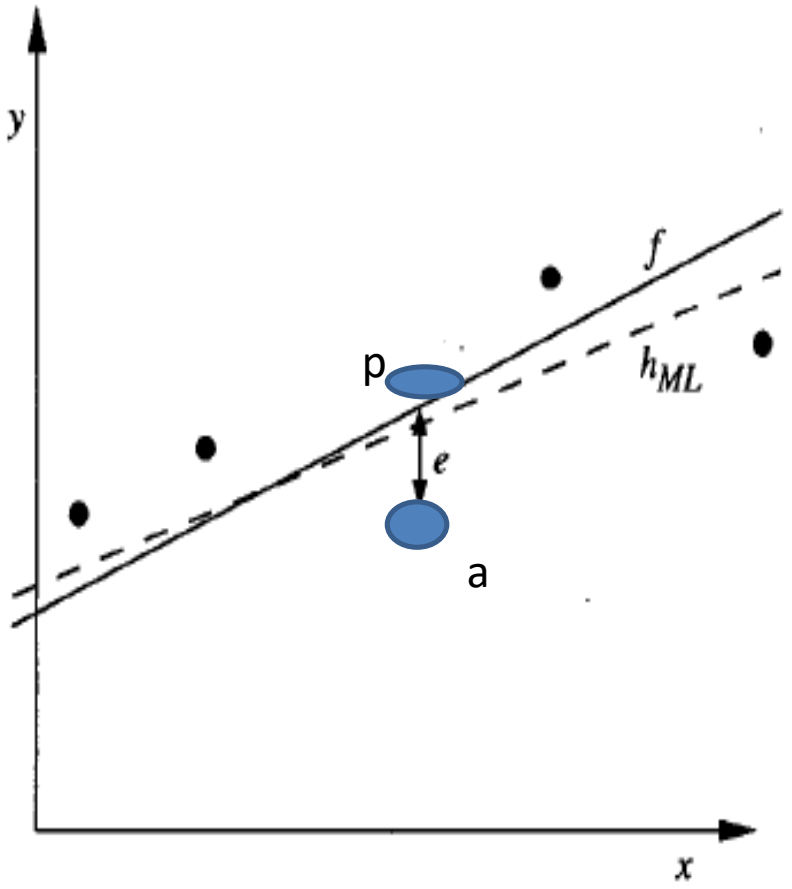
(c), the posterior probability of inconsistent hypotheses becomes zero, while posterior probabilities increase for hypotheses remaining in the version space.

6.4 Maximum Likelihood & Least-squared Error Hypotheses

Consider the following problem setting. Learner L considers an instance space X and a hypothesis space H consisting of some class of real-valued functions defined over X (i.e., each h in H is a function of the form $h : X \rightarrow \Re$, where $\Re$ represents the set of real numbers). The problem faced by L is to learn an unknown target function $f : X \rightarrow \Re$ drawn from H. A set of m training examples is provided, where the target value of each example is corrupted by random noise drawn according to a Normal probability distribution. More precisely, each training example is a pair of the form $\langle x_i, d_i \rangle$ where $d_i = f(x_i) + e_i$. Here $f(x_i)$ is the noise-free value of the target function and e_i is a random variable representing the noise. It is assumed that the values of the e_i are drawn independently and that they are distributed according to a Normal distribution with zero mean. The task of the learner is to output a maximum likelihood hypothesis, or, equivalently, a MAP hypothesis assuming all hypotheses are equally probable a priori.

A simple example of such a problem is learning a linear function, though our analysis applies to learning arbitrary real-valued functions. Figure 6.2 illustrates

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



A **linear target function f** depicted by the solid line, and a set of noisy training examples of this target function. The dashed line corresponds to the hypothesis h_{ML} *with least-squared training error, hence the maximum likelihood hypothesis*.

Notice that the maximum likelihood hypothesis is not necessarily identical to the correct hypothesis, f, because it is inferred from only a limited sample of noisy training data.

in this setting is also a maximum likelihood hypothesis, let us quickly review two basic concepts from probability theory: probability densities and Normal distributions. First, in order to discuss probabilities over continuous variables such as e, we must introduce probability *densities*. The reason, roughly, is that we wish for the total probability over all possible values of the random variable to sum to one. In the case of continuous variables we cannot achieve this by assigning a finite probability to each of the infinite set of possible values for the random variable. Instead, we speak of a probability *density* for continuous variables such as e and require that the integral of this probability density over all possible values be one. In general we will use lower case p to refer to the probability density function, to distinguish it from a finite probability P (which we will sometimes refer to as a probability *mass*). The probability density $p(x_0)$ is the limit as ϵ goes to zero, of $\frac{1}{\epsilon}$ times the probability that x will take on a value in the interval $[x_0, x_0 + \epsilon)$.

PDF for regression
PMF for classification

Probability density function:

$$p(x_0) \equiv \lim_{\epsilon \to 0} \frac{1}{\epsilon} P(x_0 \leq x < x_0 + \epsilon)$$

Second, we stated that the random noise variable e is generated by a Normal probability distribution. A Normal distribution is a smooth, bell-shaped distribution that can be completely characterized by its mean μ and its standard deviation σ. See Table 5.4 for a precise definition.

Given this background we now return to the main issue: showing that the least-squared error hypothesis is, in fact, the maximum likelihood hypothesis within our problem setting. We will show this by deriving the maximum likelihood hypothesis starting with our earlier definition Equation (6.3), but using lower case p to refer to the probability density

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

As before, we assume a fixed set of training instances $\langle x_1 \ldots x_m \rangle$ and therefore consider the data D to be the corresponding sequence of target values $D = \langle d_1 \ldots d_m \rangle$. Here $d_i = f(x_i) + e_i$. Assuming the training examples are mutually independent given h, we can write $P(D|h)$ as the product of the various $p(d_i|h)$

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

Given that the noise e_i obeys a Normal distribution with zero mean and unknown variance σ^2, each d_i must also obey a Normal distribution with variance σ^2 centered around the true target value $f(x_i)$ rather than zero. Therefore $p(d_i|h)$ can be written as a Normal distribution with variance σ^2 and mean $\mu = f(x_i)$. Let us write the formula for this Normal distribution to describe $p(d_i|h)$, beginning with the general formula for a Normal distribution from Table 5.4 and substituting the appropriate μ and σ^2. Because we are writing the expression for the probability of d_i given that h is the correct description of the target function f, we will also substitute $\mu = f(x_i) = h(x_i)$, yielding

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

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

We now apply a transformation that is common in maximum likelihood calculations: Rather than maximizing the above complicated expression we shall choose to maximize its (less complicated) logarithm. This is justified because $\ln p$ is a monotonic function of p. Therefore maximizing $\ln p$ also maximizes p.

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

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

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

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

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

Finally, we can again discard constants that are independent of h.

$$h_{ML} = \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} (d_i - h(x_i))^2 \tag{6.6}$$

6.5 MAXIMUM LIKELIHOOD HYPOTHESES FOR PREDICTING PROBABILITIES

Consider the setting in which we wish to learn a nondeterministic (probabilistic) function $f : X \rightarrow \{0, 1\}$, which has two discrete output values. For example, the instance space X might represent medical patients in terms of their symptoms, and the target function $f(x)$ might be 1 if the patient survives the disease and 0 if not. Alternatively, X might represent loan applicants in terms of their past credit history, and $f(x)$ might be 1 if the applicant successfully repays their next loan and 0 if not. In both of these cases we might well expect f to be probabilistic. For example, among a collection of patients exhibiting the same set of observable symptoms, we might find that 92% survive, and 8% do not. This unpredictability could arise from our inability to observe all the important distinguishing features of the patients, or from some genuinely probabilistic mechanism in the evolution of the disease. Whatever the source of the problem, the effect is that we have a target function $f(x)$ whose output is a probabilistic function of the input.

Given this problem setting, we might wish to learn a neural network (or other real-valued function approximator) whose output is the *probability* that $f(x) = 1$. In other words, we seek to learn the target function, $f' : X \rightarrow [0, 1]$, such that $f'(x) = P(f(x) = 1)$. In the above medical patient example, if x is one of those indistinguishable patients of which 92% survive, then $f'(x) = 0.92$ whereas the probabilistic function $f(x)$ will be equal to 1 in 92% of cases and equal to 0 in the remaining 8%.

How can we learn f' using, say, a neural network? One obvious, brute-force way would be to first collect the observed frequencies of 1's and 0's for each possible value of x and to then train the neural network to output the target frequency for each x. As we shall see below, we can instead train a neural network directly from the observed training examples of f, yet still derive a maximum likelihood hypothesis for f'.

What criterion should we optimize in order to find a maximum likelihood hypothesis for f' in this setting? To answer this question we must first obtain an expression for $P(D|h)$. Let us assume the training data D is of the form $D = \{\langle x_1, d_1 \rangle \ldots \langle x_m, d_m \rangle\}$, where d_i is the observed 0 or 1 value for $f(x_i)$.

Recall that in the maximum likelihood, least-squared error analysis of the previous section, we made the simplifying assumption that the instances $\langle x_1 \ldots x_m \rangle$ were fixed. This enabled us to characterize the data by considering only the target values d_i. 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 x_i and d_i as random variables, and assuming that each training example is drawn independently, we can write $P(D|h)$ as

$$P(D|h) = \prod_{i=1}^{m} P(x_i, d_i|h) \tag{6.7}$$

It is reasonable to assume, furthermore, that the probability of encountering any particular instance x_i is independent of the hypothesis h. For example, the probability that our training set contains a particular *patient* x_i is independent of our hypothesis about survival rates (though of course the *survival* d_i of the patient

does depend strongly on h). When x is independent of h we can rewrite the above expression (applying the product rule from Table 6.1) as

$$P(D|h) = \prod_{i=1}^{m} P(x_i, d_i|h) = \prod_{i=1}^{m} P(d_i|h, x_i) P(x_i) \tag{6.8}$$

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

$$P(d_i|h, x_i) = \begin{cases} h(x_i) & \text{if } d_i = 1 \\ (1 - h(x_i)) & \text{if } d_i = 0 \end{cases} \tag{6.9}$$

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

$$P(d_i|h, x_i) = h(x_i)^{d_i} (1 - h(x_i))^{1-d_i} \tag{6.10}$$

It is easy to verify that the expressions in Equations (6.9) and (6.10) are equivalent. Notice that when $d_i = 1$, the second term from Equation (6.10), $(1 - h(x_i))^{1-d_i}$, becomes equal to 1. Hence $P(d_i = 1|h, x_i) = h(x_i)$, which is equivalent to the first case in Equation (6.9). A similar analysis shows that the two equations are also equivalent when $d_i = 0$.

We can use Equation (6.10) to substitute for $P(d_i|h, x_i)$ in Equation (6.8) to obtain

$$P(D|h) = \prod_{i=1}^{m} h(x_i)^{d_i}(1 - h(x_i))^{1-d_i} P(x_i) \tag{6.11}$$

Now we write an expression for the maximum likelihood hypothesis

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

The last term is a constant independent of h, so it can be dropped

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} h(x_i)^{d_i}(1 - h(x_i))^{1-d_i} \tag{6.12}$$

The expression on the right side of Equation (6.12) can be seen as a generalization of the *Binomial distribution* described in Table 5.3. The expression in Equation (6.12) describes the probability that flipping each of m distinct coins will produce the outcome $\langle d_1 \ldots d_m \rangle$, assuming that each coin x_i has probability $h(x_i)$ of producing a heads. Note the Binomial distribution described in Table 5.3 is

6.8 GIBBS ALGORITHM

Although the Bayes optimal classifier obtains the best performance that can be achieved from the given training data, it can be quite costly to apply. The expense is due to the fact that it computes the posterior probability for every hypothesis in H and then combines the predictions of each hypothesis to classify each new instance.

An alternative, less optimal method is the Gibbs algorithm , defined as follows:

1. **Choose a hypothesis *h from H at random, according to the posterior probability* distribution over H.**
2. Use ***h*** *to predict the classification of the next instance x.*

Given a new instance to classify, the Gibbs algorithm simply applies a hypothesis drawn at random according to the current posterior probability distribution. Surprisingly, it can be shown that under certain conditions the expected misclassification error for the Gibbs algorithm is at most twice the expected error of the Bayes optimal classifier . More precisely, the expected value is taken over target concepts drawn at random according to the prior probability distribution assumed by the learner. Under this condition, the expected value of the error of the Gibbs algorithm is at worst twice the expected value of the error of the Bayes optimal classifier.

This result has an interesting implication for the concept learning problem described earlier. In particular, it implies that if the learner assumes a uniform prior over H, and if target concepts are in fact drawn from such a distribution when presented to the learner, *then classifying the next instance according to a hypothesis drawn at random from the current version space (according to a uniform distribution), will have expected error at most twice that of the Bayes optimal classijier. Again, we have an example where a Bayesian analysis of a* non-Bayesian algorithm yields insight into the performance of that algorithm.

6.9 NAIVE BAYES CLASSIFIER

One highly practical Bayesian learning method is the naive Bayes learner, often called the N***aive 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 (a_1, a_2.. .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 $\langle a_1, a_2 \ldots a_n \rangle$ 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

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

$$= \underset{v_j \in V}{\operatorname{argmax}} P(a_1, a_2 \ldots a_n | v_j) P(v_j) \qquad (6.19)$$

Now we could attempt to estimate the two terms in Equation (6.19) based on the training data. It is easy to estimate each of the $\boldsymbol{P(v_j)}$ *simply by counting the* frequency with which each target value *vj occurs in the training data. However,* estimating the different $P(a_1, a_2 \ldots a_n)$ *terms in this fashion is not feasible* unless we have a very, very large set of training data. The problem is that the number of these terms is equal to the number of possible instances times the number of possible target values. Therefore, we need to see every instance in the instance space many times in order to obtain reliable estimates

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:

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_i P(a_i | v_j) \tag{6.20}$$

where v_{NB} denotes the target value output by the naive Bayes classifier. Notice that in a naive Bayes classifier the number of distinct $P(a_i | v_j)$ 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 Jvj) terms are estimated, based on their frequencies over the training data. The set of these estimates corresponds to the learned hypothesis. This hypothesis is then used to classify each new instance by applying the rule in Equation (6.20). Whenever the naive Bayes assumption of conditional independence is satisfied, this naive Bayes classification VNB is identical to the MAP classification.

One interesting difference between the naive Bayes learning method and other learning methods we have considered is that there is no explicit search through the space of possible hypotheses (in this case, the space of possible hypotheses is the space of possible values that can be assigned to the various P(vj) and P(ailvj) terms). Instead, the hypothesis is formed without searching, simply by counting the frequency of various data combinations within the training examples.

6.9.1 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

$$\begin{aligned} v_{NB} &= \underset{v_j \in \{yes,no\}}{\text{argmax}} \; P(v_j) \prod_i P(a_i|v_j) \\ &= \underset{v_j \in \{yes,no\}}{\text{argmax}} \; P(v_j) \quad P(Outlook = sunny|v_j) P(Temperature = cool|v_j) \\ &\qquad \cdot P(Humidity = high|v_j) P(Wind = strong|v_j) \end{aligned} \tag{6.21}$$

Notice in the final expression that a_i has been instantiated using the particular attribute values of the new instance. To calculate v_{NB} 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 $\frac{.0206}{.0206+.0053} = .795$.

6.10 AN EXAMPLE: LEARNING TO CLASSIFY TEXT

- Naive Bayes conditional independence assumption:

$$P(doc|v_j) = \prod_{i=1}^{length(doc)} P(a_i = w_k|v_j)$$

- where $P(a_i = w_k|v_j)$ is probability that word in position i is w_k, given v_j

- One more assumption:

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

- What does this mean?
- Is this a plausible assumption?

1. Collect all words and other tokens that occur in $Examples$
   - $Vocabulary \leftarrow$ all distinct words and other tokens in $Examples$
2. Calculate the required $P(v_j)$ and $P(w_k|v_j)$ probability terms
   - For each target value v_j in V do
     - $docs_j \leftarrow$ subset of $Examples$ for which the target value is v_j
     - $P(v_j) \leftarrow \frac{|docs_j|}{|Examples|}$
     - $Text_j \leftarrow$ a single document created by concatenating all members of $docs_j$
     - $n \leftarrow$ total number of words in $Text_j$ (counting duplicate words multiple times)
     - for each word w_k in $Vocabulary$
       - $n_k \leftarrow$ number of times word w_k occurs in $Text_j$
       - $P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$

Return the estimated target value for the document *Doc*.

- *positions* $\leftarrow$ all word positions in *Doc* that contain tokens found in *Vocabulary*
- Return v_{NB}, where

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_{i \in positions} P(a_i|v_j)$$

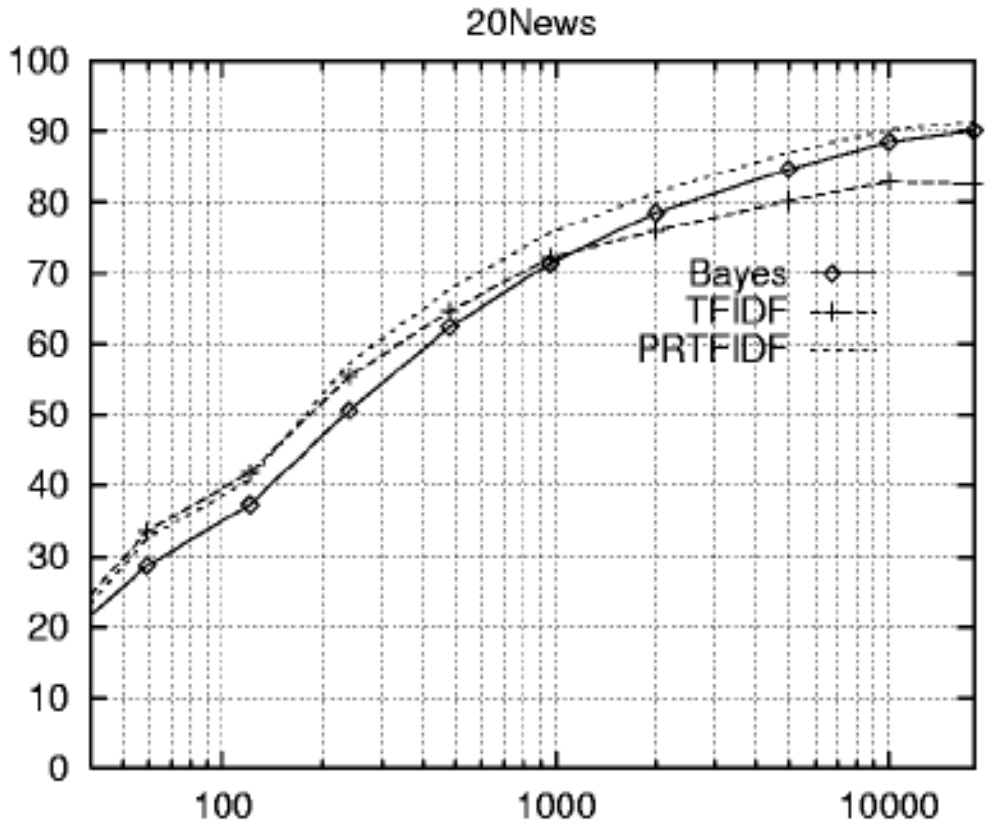
Twenty Newsgroups (Joachims, 1996)

- 1000 training documents from each of 20 groups → 20,000
- Use two third of them in learning to classify new documents according to which newsgroup it came from.

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

- Naive Bayes: 89% classification accuracy
- Random guess: ?

Learning Curve for 20 Newsgroups



Accuracy vs. Training set size (1/3 withheld for test)

(Note that the x-axis in log scale)