Bayesian Learning (Part I)

Machine Learning 6.1 - 6.6

Richard McAllister

March 31, 2008

Overview

- We can assume that quantities of interest are governed by probability distributions.
- Quantitative approach.
- Learning through direct manipulation of probabilities.
- Framework for analysis of other algorithms.

What are Bayesian learning methods doing in this book?

1. Bayesian learning algorithms are among the most practical approaches to certain types of learning problems.
2. Bayesian methods aid in understanding other learning algorithms.

Features of Bayesian Learning Methods

1. Training examples have an incremental effect on estimated probabilities of hypothesis correctness.
2. Prior knowledge and observed data combined to determine probabilities of hypotheses.
3. Hypotheses can make probabilistic predictions.
4. Combinations of multiple hypotheses can classify new instances.
5. Other methods can be measured vis a vis optimality against Bayesian methods.

The heart of the matter:

What does it mean to have the "best" hypothesis?

An answer:

best hypothesis = most probable hypothesis

Some notation:

- D: the training data
- H: the set of all hypotheses
- h: a hypothesis $h \in H$
- $P(h)$: the *prior probability* of h: the initial probability that hypothesis h holds, before we have observed the training data
- $P(D)$: the prior probability that training data D will be observed
- $P(D \mid h)$: the probability of observing data D given some world in which h holds
- $P(x \mid y)$: the probability of x occurring given that y has been observed
- $P(h \mid D)$: the *posterior probability* of h given D: the probability that h holds given the observed training data D

Bayes Theorem:

- $P(h \mid D) = \frac{P(D|h)P(h)}{P(D)}$
- This means: the posterior probability of D given h equals the probability of observing data D given some world in which h holds times the prior probability of h all over the prior probability of D.

What we're interested in:

maximum a posteriori (MAP) hypothesis: the most probable hypothesis $h \in H$ given the observed data D

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

Assuming equal probabilities for all $h \in H$ (*maximum likelihood hypothesis*):

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

Where does Bayes theorem work?

Any set H of mutually exclusive propositions whose probabilities sum to one.

An example:

A grim scenario at the doctor's office:

1. a patient has cancer
2. the patient does not have cancer

An example (cont'd):

$$P(cancer) = .008 \qquad P(\neg cancer) = .992$$
$$P(\oplus \mid cancer) = .98 \qquad P(\ominus \mid cancer) = .02$$
$$P(\oplus \mid \neg cancer) = .98 \qquad P(\ominus \mid \neg cancer) = .02$$

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

Basic Probability Formulas

- Product Rule: $P(A \wedge B) = P(A \mid B)P(B) = P(B \mid A)P(A)$
- Sum Rule: $P(A \vee B) = P(A) + P(B) - P(A \wedge B)$
- Bayes Theorem: $P(h \mid D) = \frac{P(D|h)P(h)}{P(D)}$
- Theorem of Total Probability (if events $A_1, ..., A_n$ are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1$): $P(B) = \sum_{i=1}^{n} P(B \mid A_i)P(A_i)$

The Big Idea

"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."(ML pg 158)

More Terminology (Ugh!)

- X: the instance space
- x_i: some instance from X
- D: the training data
- d_i: the target value of x_i
- c: the target concept ($c : X \rightarrow \{0, 1\}$ and $d_i = c(x_i)$)
- $\langle x_1 \ldots x_m \rangle$: the sequence of instances
- $\langle d_1 \ldots d_m \rangle$: the sequence of target values

Brute-Force MAP Learning

Two steps:

1. For each h in H, calculate the posterior probability:
$$P(h \mid D) = \frac{P(D|h)P(h)}{P(D)}$$
2. Output h_{MAP}
$$h_{MAP} \equiv \underset{h \in H}{\operatorname{argmax}} P(h \mid D)$$

Note: Since each $h \in H$ is calculated, it's possible that a lot of calculation must be done here.

Constraining Our Example

We have some flexibility in how we may choose probability distributions $P(h)$ and $P(D \mid h)$. The following assumptions have been made:

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.

These constraints imply...

- $P(h) = \frac{1}{|H|} \forall h \in H$
- $P(D \mid h) = \left\{ \begin{array}{c} 1 \text{ if } d_i = h(x_i) \forall d_i \in D \\ 0 \text{ otherwise} \end{array} \right\}$

If h is consistent with D:

We have:

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

- $| \ VS_{H,D} \ |$: the version space of H with respect to D (the subset of hypotheses from H that are consistent with D)

An unanswered question: the derivation of $P(D)$:

Given: $(\forall i \neq j)(P(h_i \wedge h_j) = 0)$ (the hypotheses are mutually exclusive):

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

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

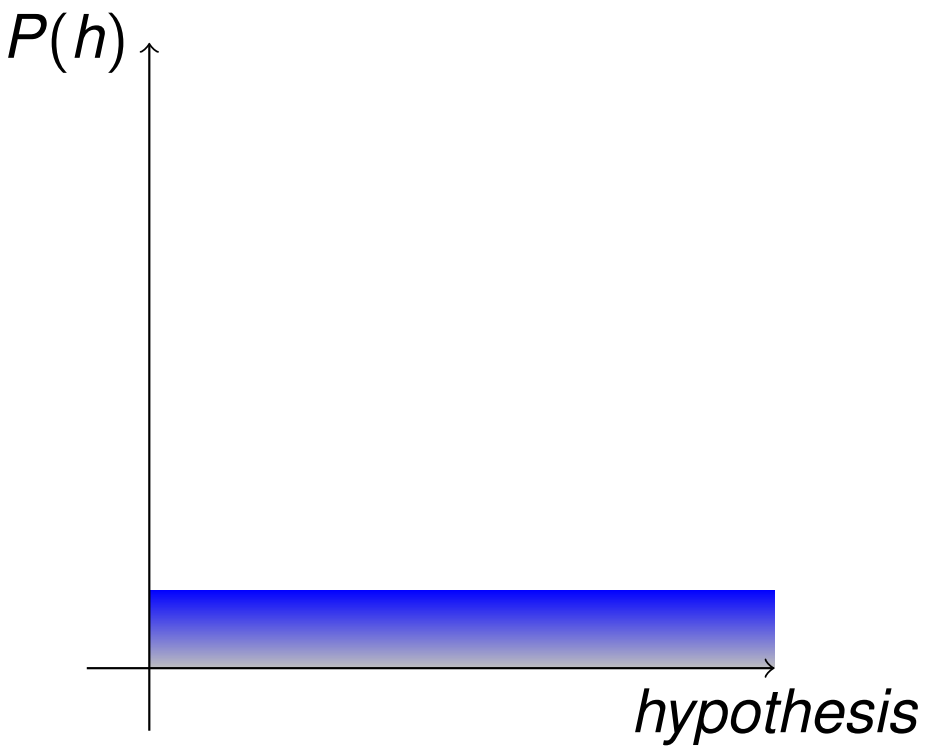
$$= \sum_{h_i \in VS_{H,D}} 1 \cdot \frac{1}{|H|}$$

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

Therefore...

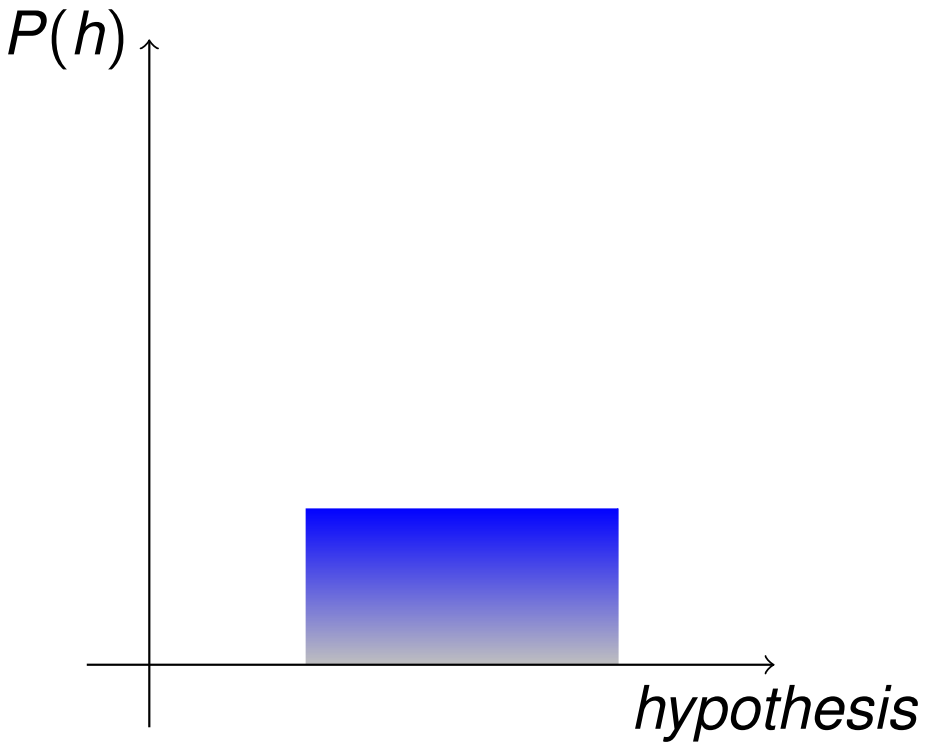
$$P(h \mid D) = \left\{ \begin{array}{c} \frac{1}{|VS_{H,D}|} \text{ if } h \text{ is consistent with } D \\ 0 \text{ otherwise} \end{array} \right\}$$

Initially: all hypotheses have same probability:



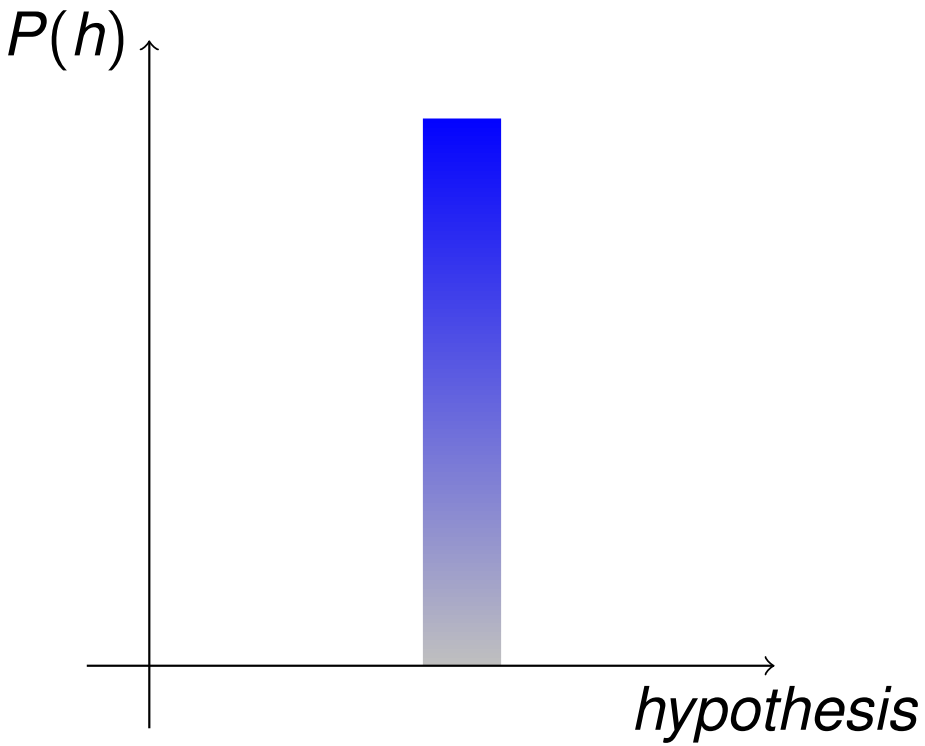
Posterior probabilities become zero for inconsistent hypotheses:

Total probability summing to 1 is shared equally among remaining hypotheses:



Posterior probabilities become zero for inconsistent hypotheses:

Total probability summing to 1 is shared equally among remaining hypotheses:



Consistent Learners

consistent learner: a learning algorithm that outputs a hypothesis that commits zero errors over the training examples.

Every consistent learner outputs a MAP hypothesis if we assume:

- uniform prior probability distribution over H
- deterministic, noise-free training data

Example: Find-S outputs the maximally specific consistent hypothesis, which is a MAP hypothesis.

Characterizing the Behavior of Learning Algorithms

Recall: the *inductive bias* of a learning algorithm is the set of assumptions B sufficient to *deductively* justify the inductive inference performed by the learner.

- inductive inference can also be modeled using probabilistic reasoning based on Bayes theorem.
- assumptions are of the form: "the prior probabilities over H are given by the distribution $P(h)$, and the strength of data in rejecting or accepting a hypothesis is given by $P(D \mid h)$."
- $P(h)$ and $P(D \mid h)$ characterize the implicit assumptions of the algorithms being studied:
  - Candidate-Elimination
  - Find-S
- $P(D \mid h)$ can also take on values other than 0 or 1.

Premise: "...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." (pg. 164)

- neural networks do this
- so do other curve fitting methods

More Terminology

- **probability density function**: $p(x_0) \equiv \lim_{\epsilon \to 0} \frac{1}{\epsilon} P(x_0 \leq x < x_0 + \epsilon)$
- **e**: a random noise variable generated by a Normal probability distribution
- $\langle x_1 \ldots x_m \rangle$: the sequence of instances (as before)
- $\langle d_1 \ldots d_m \rangle$: the sequence of target values with $d_i = f(x_i) + e_i$

We wish to show that...

...the least-squared error hypothesis is, in fact, the maximum likelihood hypothesis (within our problem setting).

- Using the previous definition of h_{ML} we have: $\underset{h \in H}{\operatorname{argmax}}\, p(D \mid h)$
- Assuming training examples are mutually independent given h:

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

- Since e_i follows a Normal distribution, d_i must also follow the same. Therefore:

  $$p(d_i \mid h) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\alpha^2}(d_i - \mu)^2}$$

  - with mean μ and variance σ^2

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

Therefore...

The maximum likelihood hypothesis h_{ML} is the one that minimizes the sum of the squared errors between the observed training values d_i and the hypothesis predictions $h(x_i)$.

Outputs and Inputs

- We have a target function $f(x)$ whose output is a probabilistic function of the input.

$$f : X \rightarrow \{0, 1\}$$

- We want a function approximator whose output is the *probability* that $f(x) = 1$

$$f' : X \rightarrow [0, 1] \text{ such that } f'(x) = P(f(x) = 1)$$

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

$$= \prod_{i=1}^{m} P(d_i \mid h, x_i) P(x_i) \text{ (applying the product rule)}$$

Therefore...

$$P(d_i \mid h, x_i) = \left\{ \begin{array}{c} h(x_i) \text{ if } d_i = 1 \\ (1 - h(x_i)) \text{ if } d_i = 0 \end{array} \right\}$$
$$= h(x_i)^{d_i}(1 - h(x_i))^{1-d_i}$$

Furthermore...

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

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

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

The right side of the last expression can be seen as a generalization of the Binomial distribution (table 5.3).

Log likelihood is used by the book:

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

Note: The above is sometimes called *cross entropy* due to it's similarity to the general form of the entropy function $-\sum_i p_i log p_i$

What we want:

- We desire a weight-training rule for neural network learning that seeks to maximize the maximum likelihood hypothesis ($G(h, D)$) using gradient ascent.

$$\begin{aligned}
\frac{\partial G(h,D)}{\partial w_{jk}} &= \sum_{i=1}^{m} \frac{\partial G(h,D)}{\partial h(x_i)} \frac{\partial h(x_i)}{\partial w_{jk}} \\
&= \sum_{i=1}^{m} \frac{\partial (d_i \ln h(x_i) + (1-d_i) \ln(1-h(x_i)))}{\partial h(x_i)} \frac{\partial h(x_i)}{\partial w_{jk}} \\
&= \sum_{i=1}^{m} \frac{d_i - h(x_i)}{h(x_i)(1-h(x_i))} \frac{\partial h(x_i)}{\partial w_{jk}}
\end{aligned}$$

Example:

Single layer of sigmoid units:

$$\frac{\partial h(x_i)}{\partial w_{jk}} = \sigma'(x_i)x_{ijk} = h(x_i)(1 - h(x_i))h_{ijk}$$

where x_{ijk} is the kth input to unit j for the ith training example, and $\sigma'(x)$ is the derivative of the sigmoid squashing function.

$$\frac{\partial G(h,D)}{\partial w_{jk}} = \sum_{i=1}^{m}(d_i - h(x_i))x_{ijk}$$

Gradient ascent:

We want to maximize $P(D \mid h)$:
On each iteration of search the weight vector is adjusted in the direction of the gradient:

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

where:

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

Gradient ascent:

Contrast this with the weight-update rule used by the Backpropagation algorithm:

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

where:

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

What the is the Minimum Description Length Principle?

- A Bayesian perspective on Occam's razor
- Motivated by interpreting the definition of h_{MAP} in the light of basic concepts from information theory.

Consider this:

$$
\begin{aligned}
h_{MAP} &= \underset{h \in H}{\operatorname{argmax}} P(D \mid h) P(h) \\
&= \underset{h \in H}{\operatorname{argmax}} log_2 P(D \mid h) + log_2 P(h) \\
&= \underset{h \in H}{\operatorname{argmin}} -log_2 P(D \mid h) - log_2 P(h)
\end{aligned}
$$

- short hypotheses are preferred.

Terminology

- C: the code used to encode a message
- i: the message
- $L_C(i)$: the description length of message i with respect to C...or more simply, the number of bits used to encode the message

Interpreting the equation $h_{MAP} = \underset{h \in H}{\text{argmin}} -log_2 P(D \mid h) - log_2 P(h)$:

- $-log_2 P(h)$: the description length of h under the optimal encoding for the hypothesis space H: $L_{C_H}(h) = -log_2 P(h)$
- $-log_2 P(D \mid h)$: the description length of the training data D given hypothesis h, under the optimal encoding fro the hypothesis space H: $L_{C_H}(D \mid h) = -log_2 P(D \mid h)$
- $h_{MAP} = \underset{h \in H}{\text{argmin}}\, L_{C_H}(h) + L_{C_H}(D \mid h)$

Minimum Description Length Principle:

Choose h_{MDL} where:

$$h_{MDL} = \underset{h \in H}{\operatorname{argmin}} L_{C_1}(h) + L_{C_1}(D \mid h)$$

- provides a way of trading off hypothesis complexity for the number of errors committed by the hypothesis.
- provides a way to deal with the issue of overfitting the data.
- short imperfect hypothesis may be selected over a long perfect hypothesis.

Thank you!