Chapter 6:
Bayesian Learning

CS 536: Machine Learning

Littman (Wu, TA)

Bayesian Learning

[Read Ch. 6, except 6.3]

[Suggested exercises: 6.1, 6.2, 6.6]

- Bayes Theorem
- MAP, ML hypotheses
- MAP learners
- Minimum description length principle
- Bayes optimal classifier
- Naive Bayes learner (if time)
- Example: Learning over text data
- Bayesian belief networks
- Expectation Maximization algorithm

Roles for Bayesian Methods

Provides practical learning algorithms:

- Naive Bayes learning
- Bayesian belief network learning
- Combine prior knowledge (prior probabilities) with observed data
- Requires prior probabilities

Provides useful conceptual framework

- Provides "gold standard" for evaluating other learning algorithms
- Additional insight into Occam's razor

Bayes Theorem

$$P(h|D) = P(D|h)\, P(h) \,/\, P(D)$$

- $P(h)$ = prior prob. of hypothesis h
- $P(D)$ = prior prob. of training data D
- $P(h|D)$ = probability of h given D
- $P(D|h)$ = probability of D given h

Choosing Hypotheses

Natural choice is most probable hypothesis given the training data, or *maximum a posteriori* hypothesis h_{MAP}:

$$h_{MAP} = \text{argmax}_{h \text{ in } H} P(h|D)$$
$$= \text{argmax}_{h \text{ in } H} P(D|h)\, P(h) \,/\, P(D)$$
$$= \text{argmax}_{h \text{ in } H} P(D|h)\, P(h)$$

If assume $P(h_i) = P(h_j)$ then can further simplify, and choose the *maximum likelihood* (ML) hypothesis

$$h_{ML} = \text{argmax}_{hi \text{ in } H} P(D|h_i)$$

Bayes Theorem

Does patient have cancer or not?

- A patient takes a lab test and the result comes back positive. The test returns a correct positive result in 98% of the cases in which the disease is actually present, and a correct negative result in 97% of the cases in which the disease is not present. Furthermore, .008 of the entire population have this cancer.

P (cancer) = P (not cancer) =

P (+|cancer) = P (–|cancer) =

P (+|not cancer) = P (–|not cancer) =

Basic Formulas for Probs

- 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:

  $P(A \vee B) = P(A) + P(B) - P(A \wedge B)$
- Theorem of total probability: if the events $A_1, \ldots, A_n$ are mutually exclusive with $\Sigma_{i=1}^{n} P(A_i) = 1$, then

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

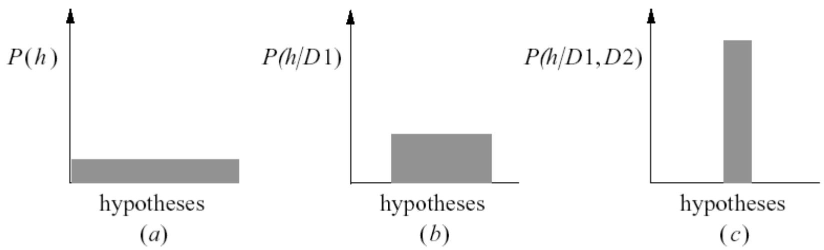
Brute Force MAP Learner

1. For each hypothesis h in H, calculate the posterior probability

   $$P(h|D) = P(D|h)\, P(h) \,/\, P(D)$$
2. Output the hypothesis h_{MAP} with the highest posterior probability

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

Evolution of Posterior Probs



- As data is added, certainty of hypotheses increases.
- What is the effect on entropy?

Real-Valued Functions

Consider any real-valued target function f

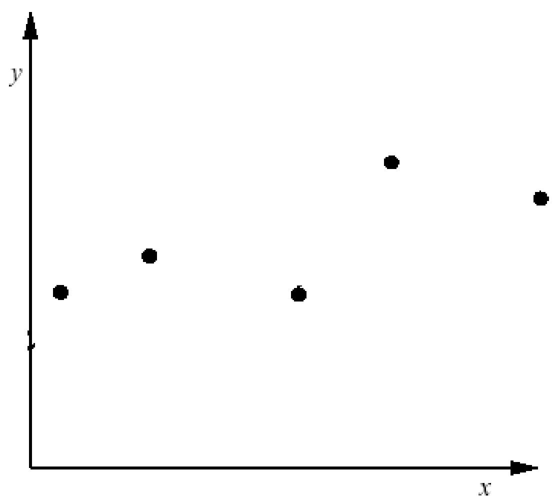
Training examples $<x_i, d_i>$, where d_i is noisy training value

- $d_i = f(x_i) + e_i$
- e_i is random variable (noise) drawn independently for each x_i 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:

$$h_{ML} = \text{argmin}_{h \text{ in } H} \Sigma_{i=1}^{m} (d_i - h(x_i))^2$$

MAP and Least Squares



MAP/Least Squares Proof

$$\begin{aligned}
h_{MAP} &= \text{argmax}_{h \text{ in } H} P(h|D) \\
&= \text{argmax}_{h \text{ in } H} P(D|h) \\
&= \text{argmax}_{h \text{ in } H} \Pi_{i=1}^{m} 1/\text{sqrt}(2\pi\sigma^2) \\
&\quad \cdot \exp(-1/2\ ((d_i - h(x_i))/\sigma)^2) \\
&= \text{argmax}_{h \text{ in } H} \Sigma_{i=1}^{m} \ln 1/\text{sqrt}(2\pi\sigma^2) \\
&\quad -1/2\ ((d_i - h(x_i))/\sigma)^2 \\
&= \text{argmax}_{h \text{ in } H} \Sigma_{i=1}^{m} -1/2\ ((d_i - h(x_i))/\sigma)^2 \\
&= \text{argmax}_{h \text{ in } H} \Sigma_{i=1}^{m} -(d_i - h(x_i))^2 \\
&= \text{argmin}_{h \text{ in } H} \Sigma_{i=1}^{m} (d_i - h(x_i))^2
\end{aligned}$$

Predicting Probabilities

Consider predicting survival probability from patient data

Training examples $<x_i, d_i>$, where d_i is 1 or 0

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

Predicting Probabilities

In this case, can show

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

Weight update rule for a sigmoid unit:

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

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

MDL Principle

Minimum Description Length Principle

Occam's razor: prefer the “shortest” hypothesis

MDL: prefer the hypothesis h that minimizes

$$h_{MDL} = \text{argmin}_{h \text{ in } H} (L_{C1}(h) + L_{C2}(D|h))$$

where $L_C(x)$ is the description length of x under encoding C

MDL Example

Example: H = decision trees, D = training data labels

- $L_{C1}(h)$ is # bits to describe tree h
- $L_{C2}(D|h)$ is # bits to describe D given h
  - Note $L_{C2}(D|h) = 0$ if examples classified perfectly by h. Need only describe exceptions.
- Hence, h_{MDL} trades off tree size for training errors

MDL Justification

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

= $\text{argmax}_{h \text{ in } H}$ ($\log_2$ P(D|h) + $\log_2$ P(h))

= $\text{argmin}_{h \text{ in } H}$ ($-\log_2$ P(D|h) $-\log_2$ P(h))

From information theory:

The optimal (shortest expected coding length) code for an event with probability p is $-\log_2 p$

So, prefer the hypothesis that minimizes

length(h) + *length*(*misclassifications*)

Classifying New Instances

So far we've sought the most probable *hypothesis* given the data D (i.e., h_{MAP})

Given new instance x, what is its most probable *classification*?

- $h_{MAP}(x)$ is not the most probable classification!

Classification Example

Consider:

- Three possible hypotheses:

$P(h_1|D) = .4$, $P(h_2|D) = .3$, $P(h_3|D) = .3$

- Given new instance x,

$h_1(x) = +$, $h_2(x) = -$, $h_3(x) = -$

- What's $h_{MAP}(x)$?
- What's most probable classification of x?

Bayes Optimal Classifier

Bayes optimal classification:

$\text{argmax}_{v_j \text{ in } V} \Sigma_{h_i \text{ in } H}$ P(v_j|h_i) P(h_i|D)

Example:

- P(h_1|D) = .4, P(−|h_1) = 0, P(+|h_2) = 1
- P(h_2|D) = .3, P(−|h_2) = 1, P(+|h_3) = 0
- P(h_3|D) = .3, P(−|h_3) = 1, P(+|h_3) = 0,

therefore

$\Sigma_{h_i \text{ in } H}$ P(+|h_i) P(h_i|D) = .4

$\Sigma_{h_i \text{ in } H}$ P(− |h_i) P(h_i|D) = .6 MAP class

Gibbs Classifier

Bayes optimal classifier provides best result, but can be expensive if many hypotheses.

Gibbs algorithm:

1. Choose one hypothesis at random, according to $P(h|D)$
2. Use this one to classify new instance

Error of Gibbs

(Not so) surprising fact: Assume target concepts are drawn at random from H according to priors on H. Then:

$$E[error_{Gibbs}] \leq 2E[error_{BayesOptimal}]$$

Suppose correct, uniform prior distribution over H, then

- Pick any hypothesis consistent with the data, with uniform probability
- Its expected error no worse than twice Bayes optimal

Naive Bayes Classifier

Along with decision trees, neural networks, *k*NN, one of the most practical and most used learning methods.

When to use:

- Moderate or large training set available
- Attributes that describe instances are conditionally independent given classification

Successful applications:

- Diagnosis
- Classifying text documents

Naive Bayes Classifier

Assume target function $f : X \rightarrow V$, where each instance x described by attributes $<a_1, a_2 \ldots a_n>$.

Most probable value of $f(x)$ is:

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

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

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

Naïve Bayes Assumption

$P(a_1, a_2 \ldots a_n | v_j) = \Pi_i P(a_i | v_j)$,

which gives

Naive Bayes classifier:

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

Note: No search in training!

Naïve Bayes Algorithm

Naïve_Bayes_Learn(*examples*)

For each target value v_j

$\hat{P}(v_j) \leftarrow$ estimate $P(v_j)$

For each attribute value a_i of each attribute a

$\hat{P}(a_i|v_j) \leftarrow$ estimate $P(a_i|v_j)$

Classify_New_Instance(*x*)

$$v_{NB} = \text{argmax}_{v_j \text{ in } V} \hat{P}(v_j) \Pi_i \hat{P}(a_i | v_j)$$

Naïve Bayes: Example

- Consider *PlayTennis* again, and new instance

<*Outlk = sun*, *Temp = cool*, *Humid = high*, *Wind = strong*>

Want to compute:

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

$P(y)\, P(sun|y)\, P(cool|y)\, P(high|y)\, P(strong|y) = .005$

$P(n)\, P(sun|n)\, P(cool|n)\, P(high|n)\, P(strong|n) = .021$

- So, $v_{NB} = n$

Naïve Bayes: Subtleties

1. Conditional independence assumption is often violated

$$P(a_1, a_2 \ldots a_n | v_j) = \Pi_i P(a_i | v_j)$$

- ...but it works surprisingly well anyway. Note don't need estimated posteriors $P(v_j|x)$ to be correct; need only that

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

- Domingos & Pazzani [1996] for analysis
- Naïve Bayes posteriors often unrealistically close to 1 or 0

Naïve Bayes: Subtleties

2. what if none of the training instances with target value v_j have attribute a_i?
   $P(a_i|v_j) = 0$, and... $P(v_j) \Pi_i P(a_i|v_j) = 0$

Solution is Bayesian estimate:

$$P(a_i|v_j) = (n_c + mp)/(n + m) \text{ where}$$

- n is number of training examples for which $v = v_j$,
- n_c number of examples for which $v = v_j$ and $a = a_i$
- p is prior estimate for $P(a_i|v_j)$
- m is weight given to prior (i.e., number of "virtual" examples)