Lecture 12: Bayesian Learning

Reading: Mitchell, Sections 6.1 - 6.10.

- ◇ Bayes Theorem
- ◇ Most likely hypotheses
- ◇ Minimum description length principle
- ◇ Bayes optimal classifier
- ◇ Naive Bayes learning

Two Roles for Bayesian Methods

1. Provides practical learning algorithms:
   - Naive Bayes learning
   - Bayesian belief network learning (this will be presented in 526B next year)

   which combine prior knowledge (prior probabilities) with observed data

2. Provides useful conceptual framework
   - Provides "gold standard" for evaluating other learning algorithms
   - Additional insight into Occam's razor

Bayes Theorem in Learning

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

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

Choosing Hypotheses

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

Generally want the most probable hypothesis given the training data

Maximum a posteriori hypothesis h_{MAP}:

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(h|D) \\ &= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \arg\max_{h \in H} P(D|h)P(h) \end{aligned}$$

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

$$h_{ML} = \arg\max_{h_i \in H} P(D|h_i)$$

Basic Formulas for Probabilities

- *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 events $A_1, \ldots, A_n$ are mutually exclusive with $\Sigma_{i=1}^{n} P(A_i) = 1$, then

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

Example: Using 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 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. Furthermore, $.008$ of the entire population have this cancer.

$$P(cancer) = \qquad P(\neg cancer) =$$
$$P(+|cancer) = \qquad P(-|cancer) =$$
$$P(+|\neg cancer) = \qquad P(-|\neg cancer) =$$

$$P(cancer|+) =$$

Brute Force MAP Hypothesis Learner

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

$$h_{MAP} = \arg\max_{h \in H} P(h|D)$$

Relation to Concept Learning

Consider our usual concept learning task: instance space X, hypothesis space H, training examples D. What would Bayes rule produce as the MAP hypothesis?

Assume a fixed set of instances $\langle x_1, \ldots, x_m \rangle$ with classifications $\langle c(x_1), \ldots, c(x_m) \rangle$.

Choose $P(D|h)$:

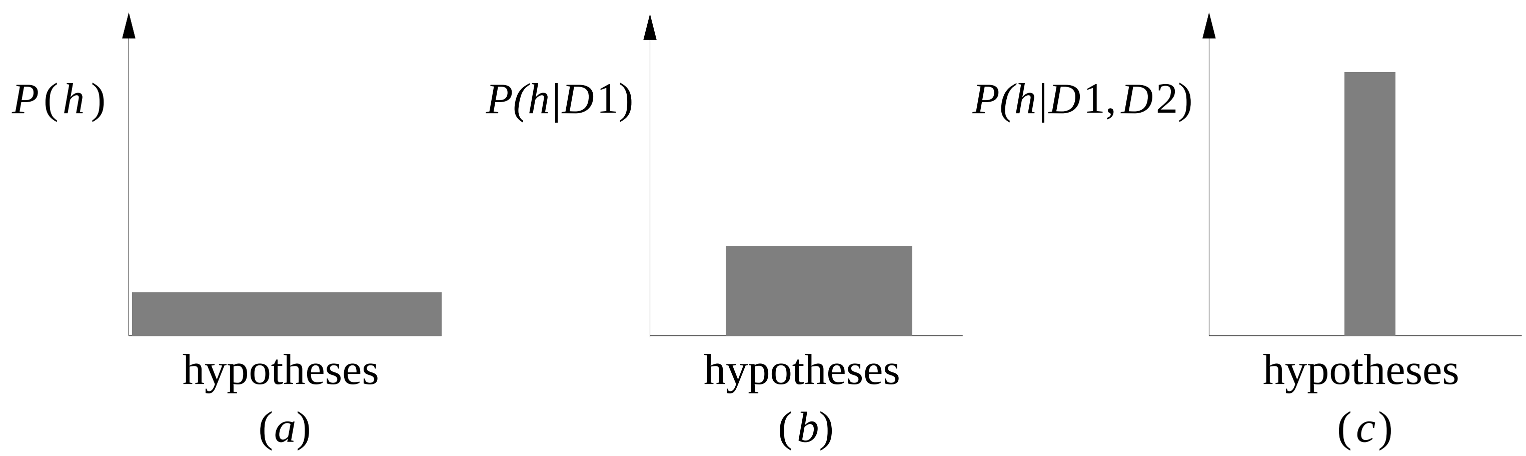
$$P(D|H) = \begin{cases} 1 & \text{if } h \text{ consistent with} D \\ 0 & \text{otherwise} \end{cases}$$

Choose $P(h)$ to be *uniform* distribution: $P(h) = \frac{1}{|H|}$ for all h in H

Then:

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

Evolution of Posterior Probabilities



Learning A Real Valued Function

Consider any real-valued target function f

The training examples are $\langle x_i, d_i \rangle$, where d_i is noisy the noisy target value:

$$d_i = f(x_i) + e_i,$$

where 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:

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

How can we show this?

Learning A Real Valued Function

$$
\begin{aligned}
h_{ML} &= \arg\max_{h \in H} p(D|h) \\
&= \arg\max_{h \in H} \prod_{i=1}^{m} p(d_i|h) \text{ (because the data points are independent)} \\
&= \arg\max_{h \in H} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{d_i - h(x_i)}{\sigma})^2} \text{ (because the noise is Gaussian)}
\end{aligned}
$$

Maximize natural log of this instead... basic idea used when we deal with products

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

Learning to Predict Probabilities

Consider predicting survival probability from patient data $\langle x_i, d_i \rangle$, where d_i is 1 or 0

We want to train neural network to output a $probability$ given x_i (not a 0 or 1)

The brute-force approach would be to estimate the probabilities from the data and then train the network using them... but we want to avoid that.

We will do an analysis of the most likely hypothesis, similar to the previous one.

Analysis

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

Since h is our hypothesis about the probability of each classification:

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

$$= h(x_i)^{d_i}(1 - h(x_i))^{1-d_i}$$

The ML hypothesis is:

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

The last factor is a constant independent of h so it can be dropped.

And by taking logs, like before, we have:

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

Maximizing Likelihood with a Neural Net

We want to maximize the likelihood of a hypothesis G(h,D):

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

Weight update rule for a sigmoid unit:

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

where

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

Minimum Description Length Principle (MDL)

Occam's razor: prefer the shortest hypothesis

MDL: prefer the hypothesis h that minimizes

$$h_{MDL} = \arg\min_{h \in H} L_{C_1}(h) + L_{C_2}(D|h)$$

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

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

- $L_{C_1}(h)$ is # bits to describe tree h
- $L_{C_2}(D|h)$ is # bits to describe D given h
  - Note $L_{C_2}(D|h) = 0$ if examples classified perfectly by h. Need only describe exceptions
- Hence h_{MDL} trades off tree size for training errors

Minimum Description Length Principle

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(D|h)P(h) \\ &= \arg\max_{h \in H} \log_2 P(D|h) + \log_2 P(h) \\ &= \arg\min_{h \in H} -\log_2 P(D|h) - \log_2 P(h) \qquad (1) \end{aligned}$$

We know from information theory that the optimal (shortest expected coding length) code for an event with probability p is $-\log_2 p$ bits.

So we can interpret (1) as follows:

- $-\log_2 P(h)$ is length of h under optimal code
- $-\log_2 P(D|h)$ is length of D given h under optimal code

So according to MDL, we prefer the hypothesis that minimizes

$$length(h) + length(misclassifications)$$

Most Probable Classification of New Instances

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

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

$h_{MAP}(x)$ (called the *Naive Bayes classification* is **NOT** the most probable classification!

Example:

Consider three possible hypotheses:

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

Given a new instance x,

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

What is the most probable classification of x?

Bayes Optimal Classifier

Bayes optimal classification:

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

In our example:

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

Therefore

$$\sum_{h_i \in H} P(+|h_i)P(h_i|D) \; = \; .4$$
$$\sum_{h_i \in H} P(-|h_i)P(h_i|D) \; = \; .6$$

and the most probably classification is $-$.

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 to classify new instance

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 from VS, with uniform probability
- Its expected error no worse than twice Bayes optimal!

Naive Bayes Classifier

Along with decision trees, neural networks, nearest neighbor, it is one of the most practical learning methods!

When to use it:

- A moderate or large training set is available (need enough data to get reliable probability estimates)
- The attributes that describe the instances are conditionally independent given the classification

Successful applications:

- Diagnosis (medical and other)
- Classifying text documents

Naive Bayes Classifier

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

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

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

Naive Bayes assumption:

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

which gives

Naive Bayes classifier: $v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$

Naive Bayes Algorithm

Naive_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 $P(a_i|v_j)$

It is easy to estimate these probabilities just by counting!

Classify_New_Instance(x)

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

Naive Bayes: Example

Consider $PlayTennis$ again, and new instance

$$\langle Outlk = sun, Temp = cool, Humid = high, Wind = strong \rangle$$

Want to compute:

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

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

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

$$\rightarrow v_{NB} = n$$

Naive Bayes: Subtleties

1. Conditional independence assumption is often violated

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

   But it works surprisingly well anyway! Note that we do not need the estimated posteriors $\hat{P}(v_j|x)$ to be correct; we need only that

   $$\arg\max_{v_j \in V} \hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = \arg\max_{v_j \in V} P(v_j) P(a_1 \ldots, a_n | v_j)$$

   Naive Bayes posteriors are often unrealistically close to 1 or 0

2. What if none of the training instances with target value v_j have attribute value a_i? Then

   $\hat{P}(a_i|v_j) = 0$, and... $\hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = 0$

Typical solution is Bayesian estimate for $\hat{P}(a_i|v_j)$

$$\hat{P}(a_i|v_j) \leftarrow \frac{n_c + mp}{n + m}$$

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 $\hat{P}(a_i|v_j)$
- m is weight given to prior (i.e. number of "virtual" examples)

Learning to Classify Text

Why?

- Learn which news articles are of interest
- Learn to classify web pages by topic

Naive Bayes is among most effective algorithms

What attributes shall we use to represent text documents?

Learning to Classify Text

Target concept $Interesting? : Document \rightarrow \{+, -\}$

1. Represent each document by vector of words: one attribute per word position in document
2. Learning: Use training examples to estimate
   - $P(+)$
   - $P(-)$
   - $P(doc|+)$
   - $P(doc|-)$

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

Naive Bayes Learning for Text

Input: $Examples$ (the set of documents), V (the appropriate classifications)

1. Collect all words and other tokens that occur in $Examples$ into a $Vocabulary$
2. calculate the required $P(v_j)$ and $P(w_k|v_j)$ probability terms, as follows: 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 $Text_j$
     - $P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$

Using the Naive Bayes Classifier

Input: a new document Doc]

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

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

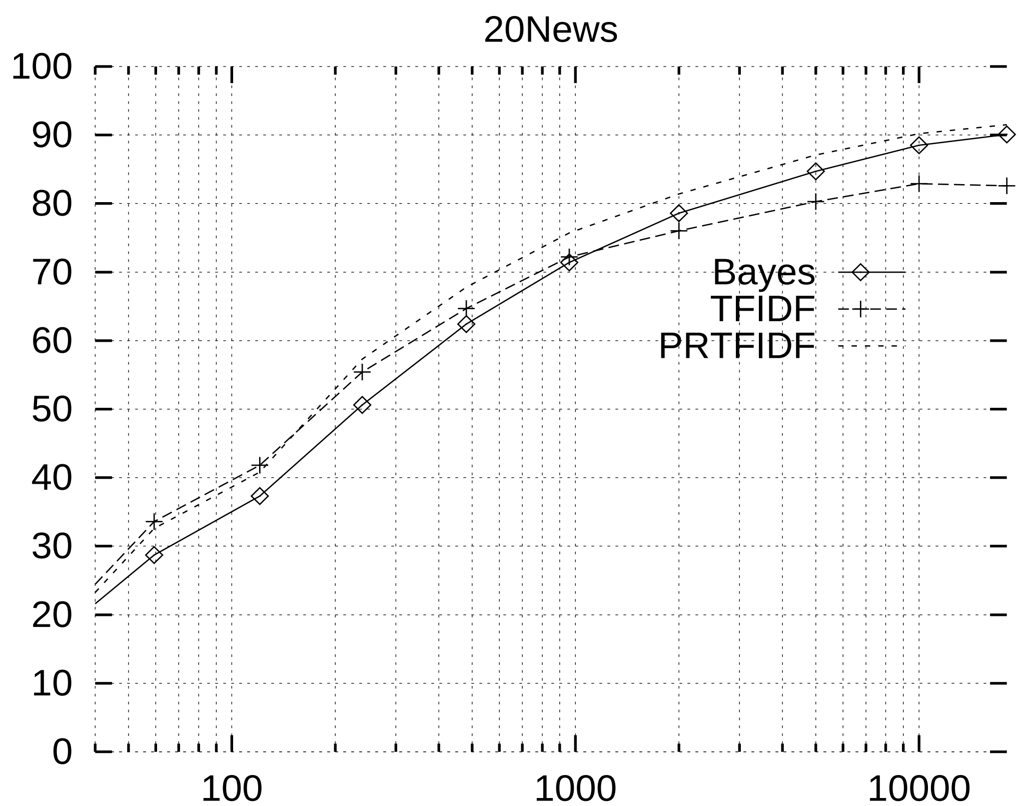
Twenty NewsGroups

Given 1000 training documents from each group, learn to classify new documents according to which newsgroup they came from

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

Learning Curve for 20 Newsgroups



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