Bayesian Approach

What should h be?

- A collection of possible predictions, or
- A collection of functions that map the data x to a possible prediction y

Most Probable Classification of New Instances

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

But the most important question is: given new instance x, what is its most probable *classification*?

- Although it may seem that h_{MAP} can answer this question, but in fact we can do better
- $h_{MAP}(x)$ is not necessarily the most probable classification! (if we are dealing with multiple hypotheses supporting each class)

Most Probable Classification of New Instances

Consider:

- Three possible hypotheses:

$$P(h_1|D) = 0.4, \qquad P(h_2|D) = 0.3, \qquad P(h_3|D) = 0.3$$

- Given new instance x,

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

- What's most probable classification of x?

Bayes Optimal Classifier

Bayes optimal classification:

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

Example:

$$P(h_1|D) = 0.4, \quad P(-|h_1) = 0, \quad \mathrm{P}(+|h_1) = 1$$

$$P(h_2|D) = 0.3, \quad P(-|h_2) = 1, \quad \mathrm{P}(+|h_2) = 0$$

$$P(h_3|D) = 0.3, \quad P(-|h_3) = 1, \quad \mathrm{P}(+|h_3) = 0$$

Bayes Optimal Classifier

therefore

$$\sum_{h_i \in H} P(+|\, h_i)\mathrm{P}(h_i|D) = 0.4$$

$$\sum_{h_i \in H} P(-|\, h_i)\mathrm{P}(h_i|D) = 0.6$$

and

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j|\, h_i)P(h_i|\mathrm{D}) = \quad -$$

v_j is the class value from the set of possible class values ($v_j \in V$)

Bayes Optimal Classifier

The Bayes optimal classifier is a probabilistic model that makes the most probable prediction for a new sample based on the training data. This is different than MAP inference.

- In MAP framework, we seek the most probable hypothesis, among the space of hypothesis.
- But in Bayesian optimal classification, the most probable classification of the new instance is obtained by combining the predictions of ALL hypotheses (in the hypothesis space), weighted by their posterior probabilities:

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

The optimal classification of the new instance is the value v_j, for which $P(v_j|D)$ is maximum

Bayes Optimal Classifier

It can be mathematically shown that no other classification method using the same hypothesis space and same prior knowledge can outperform the Bayes Optimal Classifier method on average.

胜过

Bayes Optimal Classifier

Why do we need any other methods?

Bayes Optimal Classifier

The Bayes rule depends on unknown quantities, so we need to use the data to find some approximation of those quantities.

吉布斯分类

Gibbs Classifier

Bayes optimal classifier is very inefficient.

An alternative, less optimal method is Gibbs algorithm.

Gibbs algorithm:

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

Surprising fact: Assuming target concepts are drawn at random from H $(h \in H)$ according to priors on H. Then

$$E[error_{Gibbs}] \leq 2 \times E[error_{Bayes\ Optimal}]$$

Naive Bayes Classifier

Along with decision trees, neural networks, nearest neighbour, one of the most practical learning methods.

When to use

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

Successful applications:

- Classifying text documents
- Gaussian Naive Bayes for real-valued data

Naive Bayes Classifier

Assume target function $f : X \rightarrow V$, where each instance x described by attributes $\langle x_1, x_2, \ldots, x_n \rangle$.

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

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j | x_1, x_2, \ldots, x_n)$$

$$v_{MAP} = \arg\max_{v_j \in V} \frac{P(x_1, x_2, \ldots, x_n | v_j) P(v_j)}{P(x_1, x_2, \ldots, x_n)}$$

$$= \arg\max_{v_j \in V} P(x_1, x_2, \ldots, x_n | v_j) \mathrm{P}(v_j)$$

Naive Bayes Classifier

Naive Bayes assumption:

$$P(x_1, x_2, \ldots, x_n | v_j) = \prod_i P(x_i | v_j)$$

- Attributes are statistically independent (given the class value)
  - which means knowledge about the value of a particular attribute tells us nothing about the value of another attribute (if the class is known)

Which gives **Naive Bayes classifier**:

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

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 x_i:

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

Classify New Instance (for sample x)

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

Naive Bayes Example: *PlayTennis*

outlook	temperature	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Naive Bayes Example: *PlayTennis*

What are the required probabilities to predict *PlayTennis*?

Outlook			Temperature			Humidity			Windy		
	Yes	No		Yes	No		Yes	No		Yes	No
Sunny	2	3	Hot	2	2	High	3	4	False	6	2
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3
Rainy	3	2	Cool	3	1						
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5
Rainy	3/9	2/5	Cool	3/9	1/5						
Play											
Yes	No										
9	5										
9/14	5/14										

Naive Bayes Example: *PlayTennis*

Say we have the new instance:

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

We want to compute:

$$v_{NB} = arg \max_{v_j \in \{\text{“}yes\text{”},\text{”}no\text{”}\}} P(v_j) \prod_i P(x_i | v_j)$$

Naive Bayes Example: *PlayTennis*

So we first calculate the likelihood of the two classes, “yes” and “no”

For “yes” $= P(\text{“}yes\text{”}) \times P(sun|\text{”}yes\text{”}) \times P(cool|\text{”}yes\text{”}) P(high|\text{”}yes\text{”}) \times P(true|\text{”}yes\text{”})$

$$0.0053 = \frac{9}{14} \times \frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9}$$

For “no” $= P(\text{“}no\text{”}) \times P(sun|\text{”}no\text{”}) \times P(cool|\text{”}no\text{”}) P(high|\text{”}no\text{”}) \times P(true|\text{”}no\text{”})$

$$0.0206 = \frac{5}{14} \times \frac{3}{5} \times \frac{1}{5} \times \frac{4}{9} \times \frac{3}{9}$$

Naive Bayes Example: PlayTennis

Then convert to a probability by normalisation

$$P(\text{“}yes\text{”}) = \frac{0.0053}{(0.0053 + 0.0206)} = 0.205$$

$$P(\text{“}no\text{”}) = \frac{0.0206}{(0.0053 + 0.0206)} = 0.795$$

The Naive Bayes classification is “no”.

Naive Bayes: Subtleties

Conditional independence assumption is often violated

$$P(x_1, x_2, \ldots, x_n | v_j) = \prod_i P(x_i | v_j)$$

– …but it works surprisingly well anyway. Note that you don't need the estimated posteriors $\hat{P}(v_j|x)$ to be correct; You need only that

$$arg \max_{v_j \in V} \hat{P}(v_j) \prod_i \hat{P}(x_i|v_j) = \arg \max_{v_j \in V} P(v_j) P(x_1, x_2, \ldots, x_n | v_j)$$

i.e. maximum probability is assigned to correct class

- see [Domingos & Pazzani, 1996] for analysis
- Naive Bayes posteriors often unrealistically close to 1 or 0
- adding too many redundant attributes will cause problems (e.g. identical attributes)

Naive Bayes: "zero-frequency" problem

What if none of the training instances with target value v_j have attribute value x_i? Then

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

$$\hat{P}(v_j)\prod_i \hat{P}(x_i|v_j) = 0$$

Pseudo-counts add 1 to each count (a version of the *Laplace Estimator*)

(In some cases adding a constant different from 1 might be more appropriate)

Naive Bayes: numeric attributes

- Usual assumption: attributes have a *normal* or *Gaussian* probability distribution (given the class)
- The *probability density function* for the normal distribution is defined by two parameters:
  - The sample mean μ:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

  - The standard deviation σ:

$$\sigma = \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \mu)^2$$

These parameters have to be defined for each class separately.

Naive Bayes: numeric attributes

Then we have the density function $f(x)$:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Example: continuous attribute *temperature* with $mean = 73$ and $standard\ deviation = 6.2$. Density value

$$f(temperature = 66|"yes") = \frac{1}{\sqrt{2\pi}6.2} e^{-\frac{(66-73)^2}{2\times 6.2^2}} = 0.0340$$

Missing values during training are not included in calculation of mean and standard deviation.

Categorical random variables

Categorical variables or features (also called discrete or nominal) are ubiquitous in machine learning. For example in text classification.

- Perhaps the most common form of the Bernoulli distribution models whether or not a word occurs in a document. That is, for the i-th word in our vocabulary we have a random variable X_i governed by a Bernoulli distribution. The joint distribution over the bit vector $X = (X_1, \ldots, X_k)$ is called a *multivariate Bernoulli distribution*.

- Variables with more than two outcomes are also common: for example, every word position in an e-mail corresponds to a categorical variable with k outcomes, where k is the size of the vocabulary. The multinomial distribution manifests itself as a count vector: a histogram of the number of occurrences of all vocabulary words in a document. This establishes an alternative way of modelling text documents that allows the number of occurrences of a word to influence the classification of a document.

Categorical random variables

Both these document models are in common use. Despite their differences, they both assume independence between word occurrences, generally referred to as the naive Bayes assumption.

- In the multinomial document model, this follows from the very use of the multinomial distribution, which assumes that words at different word positions are drawn independently from the same categorical distribution.
- In the multivariate Bernoulli model we assume that the bits in a bit vector are statistically independent, which allows us to compute the joint probability of a particular bit vector $(x_1, \ldots, x_k)$ as the product of the probabilities of each component $P(X_i = x_i)$.
- In practice, such word independence assumptions are often not true: if we know that an e-mail contains the word 'Viagra', we can be quite sure that it will also contain the word 'pill'. Violated independence assumptions reduce the quality of probability estimates but may still allow good classification performance.

Example application: Learning to Classify Text

In machine learning, the classic example of applications of Naive Bayes is learning to classify text documents.

Here is a simplified version in the multinomial document model.

Learning to Classify Text

For example:

- 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
    - $P(+)$
    - $P(-)$
    - $P(doc|+)$
    - $P(doc|-)$

Learning to Classify Text

Naive Bayes conditional independence assumption

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

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

one more assumption:

$$P(x_i = w_k|v_j) = P(x_m = w_k|v_j), \forall i, m$$

“bag of words”

Application: 20 Newsgroups

Given: 1000 training documents from each group
Learning task: classify each new document by newsgroup it 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	

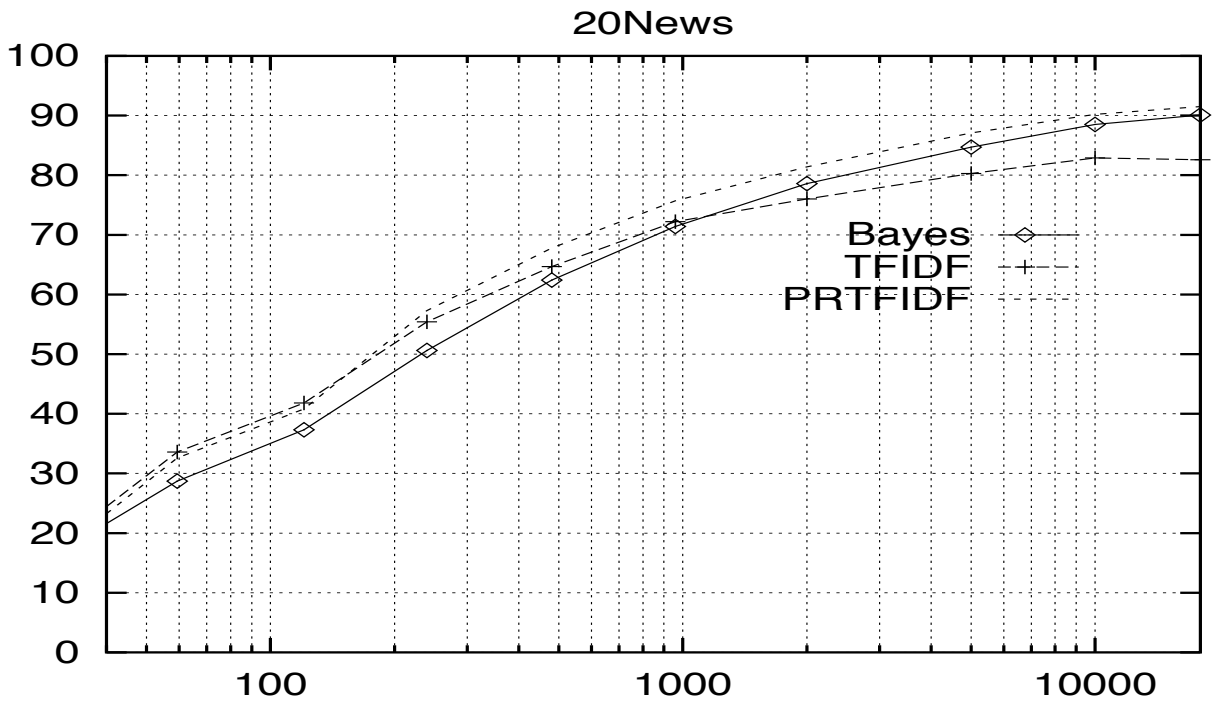
Naive Bayes: 89% classification accuracy

Article from rec.sport.hockey

```
Path: cantaloupe.srv.cs.cmu.edu!das-news.harvard.edu!ogicse!uwm.edu
From: xxx@yyy.zzz.edu (John Doe)
Subject: Re: This year's biggest and worst (opinion)...
Date: 5 Apr 93 09:53:39 GMT

I can only comment on the Kings, but the most obvious candidate
for pleasant surprise is Alex Zhitnik. He came highly touted as
a defensive defenseman, but he's clearly much more than that.
Great skater and hard shot (though wish he were more accurate).
In fact, he pretty much allowed the Kings to trade away that
huge defensive liability Paul Coffey. Kelly Hrudey is only the
biggest disappointment if you thought he was any good to begin
with. But, at best, he's only a mediocre goaltender. A better
choice would be Tomas Sandstrom, though not through any fault
of his own, but because some thugs in Toronto decided ...
```

Learning Curve for 20 Newsgroups



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

Probabilistic decision rules

We again use the example of Naive Bayes for text classification to illustrate, using both the multinomial and multivariate Bernoulli models.

Training a naive Bayes model

Consider the following e-mails consisting of five words a, b, c, d, e:

$e1$: $b\ d\ e\ b\ b\ d\ e$	$e5$: $a\ b\ a\ b\ a\ b\ a\ e\ d$
$e2$: $b\ c\ e\ b\ b\ d\ d\ e\ c\ c$	$e6$: $a\ c\ a\ c\ a\ c\ a\ e\ d$
$e3$: $a\ d\ a\ d\ e\ a\ e\ e$	$e7$: $e\ a\ e\ d\ a\ e\ a$
$e4$: $b\ a\ d\ b\ e\ d\ a\ b$	$e8$: $d\ e\ d\ e\ d$

We are told that the e-mails on the left are spam and those on the right are ham, and so we use them as a small training set to train our Bayesian classifier.

- First, we decide that d and e are so-called stop words that are too common to convey class information.
- The remaining words, a, b and c, constitute our vocabulary.

Training data for naive Bayes

The same data set described by bit vectors (presence of words).

We can use Bernoulli model.

E-mail	$a?$	$b?$	$c?$	Class
e_1	0	1	0	$+$
e_2	0	1	1	$+$
e_3	1	0	0	$+$
e_4	1	1	0	$+$
e_5	1	1	0	$-$
e_6	1	0	1	$-$
e_7	1	0	0	$-$
e_8	0	0	0	$-$

Training data for naive Bayes

A small e-mail data set described by count vectors.

We can use multinomial model.

E-mail	$\#a$	$\#b$	$\#c$	Class
e_1	0	3	0	$+$
e_2	0	3	3	$+$
e_3	3	0	0	$+$
e_4	2	3	0	$+$
e_5	4	3	0	$-$
e_6	4	0	3	$-$
e_7	3	0	0	$-$
e_8	0	0	0	$-$

Training a naive Bayes model

In the multivariate Bernoulli model e-mails are represented by bit vectors, as before.

- Adding the bit vectors for each class results in $(2, 3, 1)$ for spam and $(3, 1, 1)$ for ham.
- Each count is to be divided by the number of documents in a class, in order to get an estimate of the probability of a document containing a particular vocabulary word.
- Probability smoothing now means adding two pseudo-documents, one containing each word and one containing none of them.
- This results in the estimated parameter vectors

$$\hat{\theta}^{\oplus} = \left(\frac{3}{6}, \frac{4}{6}, \frac{2}{6}\right) = (0.5, 0.67, 0.33) \quad \text{for spam}$$

$$\hat{\theta}^{\ominus} = \left(\frac{4}{6}, \frac{2}{6}, \frac{2}{6}\right) = (0.67, 0.33, 0.33) \quad \text{for ham}$$

Training a naive Bayes model

For the multinomial model, we represent each e-mail as a count vector, as before.

- In order to estimate the parameters of the multinomial, we sum up the count vectors for each class, which gives $(5, 9, 3)$ for spam and $(11, 3, 3)$ for ham.
- To smooth these probability estimates we add one pseudo-count for each vocabulary word, which brings the total number of occurrences of vocabulary words to 20 for each class.
- The estimated parameter vectors are thus

$$\hat{\theta}^{\oplus} = \left(\frac{6}{20}, \frac{10}{20}, \frac{4}{20}\right) = (0.3, 0.5, 0.2) \quad \text{for spam}$$

$$\hat{\theta}^{\ominus} = \left(\frac{12}{20}, \frac{4}{20}, \frac{4}{20}\right) = (0.6, 0.2, 0.2) \quad \text{for ham}$$

Prediction using a naive Bayes model

Suppose our vocabulary contains three words a, b and c, and we use a multivariate Bernoulli model for our e-mails, with parameters

$$\theta^{\oplus} = (0.5, 0.67, 0.33) \qquad \theta^{\ominus} = (0.67, 0.33, 0.33)$$

This means, for example, that the presence of b is twice as likely in spam (+), compared with ham (−).

The e-mail to be classified contains words a and b but not c, and hence is described by the bit vector $x = (1, 1, 0)$. We obtain likelihoods

$$P(x|\oplus) = 0.50 \times 0.67 \times (1 - 0.33) = 0.222$$

$$P(x|\ominus) = 0.67 \times 0.33 \times (1 - 0.33) = 0.148$$

The ML classification of x is thus spam.

Prediction using a naive Bayes model

In the case of two classes it is often convenient to work with likelihood ratios and odds.

- The likelihood ratio can be calculated as

$$\frac{P(x|\oplus)}{P(x|\ominus)} = \frac{0.5}{0.67}\frac{0.67}{0.33}\frac{(1-0.33)}{(1-0.33)} = \frac{3}{2} > 1$$

- This means that the MAP classification of x is also spam if the prior odds are more than $2/3$, but ham if they are less than that.
- For example, with 33% spam and 67% ham the prior odds are $\frac{P(\oplus)}{P(\ominus)} = \frac{0.33}{0.67} = \frac{1}{2}$, resulting in a posterior odds of

$$\frac{P(\oplus|x)}{P(\ominus|x)} = \frac{P(x|\oplus)}{P(x|\ominus)}\frac{P(\oplus)}{P(\ominus)} = \frac{3}{2}\times\frac{1}{2} = \frac{3}{4} < 1$$

In this case the likelihood ratio for x is not strong enough to push the decision away from the prior

Prediction using a naive Bayes model

Alternatively, we can employ a multinomial model. The parameters of a multinomial establish a distribution over the words in the vocabulary, say

$$\theta^{\oplus} = (0.3,0.5,0.2) \qquad \theta^{\ominus} = (0.6,0.2,0.2)$$

The e-mail to be classified contains three occurrences of word a, one single occurrence of word b and no occurrences of word c, and hence is described by the count vector $x = (3,1,0)$. The total number of vocabulary word occurrences is $n = 4$. We obtain likelihoods:

$$P(x|\oplus) = 4!\frac{0.3^3}{3!}\frac{0.5^1}{1!}\frac{0.2^0}{0!} = 0.054$$

$$P(x|\ominus) = 4!\frac{0.6^3}{3!}\frac{0.2^1}{1!}\frac{0.2^0}{0!} = 0.1728$$

The likelihood ratio is $(\frac{0.3}{0.6})^3(\frac{0.5}{0.2})^1(\frac{0.2}{0.2})^0 = \frac{5}{16}$. The ML classification of x is thus ham, the opposite of the multivariate Bernoulli model. This is mainly because of the three occurrences of word a, which provide strong evidence for ham.

Naive Bayes: missing values

What about missing values ? A very basic approach is:

- Training: instance is not included in frequency count for attribute value-class combination
- Classification: attribute will be omitted from calculation

Can we do better ?

Missing values

Let's use the spam-ham data in slide 9. Suppose we skimmed an e-mail and noticed that it contains the word 'lottery' but we haven't looked closely enough to determine whether it uses the word 'Viagra'. This means that we don't know whether to use the second or the fourth row (in slide 9) to make a prediction. This is a problem, as we would predict spam if the e-mail contained the word 'Viagra' (second row) and ham if it didn't (fourth row).

The solution is to average these two rows, using the probability of 'Viagra' occurring in any e-mail (spam or not):

$$P(Y|lottery) = P(Y|Viagra = 0, lottery)P(Viagra = 0) \\ +P(Y|Viagra = 1, lottery)P(Viagra = 1)$$

Missing values

For instance, suppose for the sake of argument that one in ten e-mails contain the word 'Viagra', then $P(Viagra = 1) = 0.10$ and $P(Viagra = 0) = 0.90$. Using the above formula, we obtain

$$P(Y = spam|lottery = 1) = 0.65\times0.90 + 0.40\times0.10 = 0.625$$

and

$$P(Y = ham|lottery = 1) = 0.35\times0.90 + 0.60\times0.10 = 0.375$$

Because the occurrence of 'Viagra' in any e-mail is relatively rare, the resulting distribution deviates only a little from the second row in the data.

Likelihood ratio

As a matter of fact, statisticians work very often with different conditional probabilities, given by the likelihood function $P(X|Y)$.

- I like to think of these as thought experiments: if somebody were to send me a spam e-mail, how likely would it be that it contains exactly the words of the e-mail I'm looking at? And how likely if it were a ham e-mail instead?
- What really matters is not the magnitude of these likelihoods, but their ratio: how much more likely is it to observe this combination of words in a spam e-mail than it is in a non-spam e-mail.
- For instance, suppose that for a particular e-mail described by X we have $P(X|Y = spam) = 3.5\times10^{-5}$ and $P(X|Y = ham) = 7.4\times10^{-6}$, then observing X in a spam e-mail is nearly five times more likely than it is in a ham e-mail.
- This suggests the following decision rule: predict spam if the likelihood ratio is larger than 1 and ham otherwise.

When to use likelihoods

Use likelihoods if you want to ignore the prior distribution or assume it uniform, and posterior probabilities otherwise.

Posterior odds

$$\frac{P(Y = spam|Viagra = 0, lottery = 0)}{P(Y = ham|Viagra = 0, lottery = 0)} = \frac{0.31}{0.69} = 0.45$$

$$\frac{P(Y = spam|Viagra = 1, lottery = 1)}{P(Y = ham|Viagra = 1, lottery = 1)} = \frac{0.40}{0.60} = 0.67$$

$$\frac{P(Y = spam|Viagra = 0, lottery = 1)}{P(Y = ham|Viagra = 0, lottery = 1)} = \frac{0.65}{0.35} = 1.9$$

$$\frac{P(Y = spam|Viagra = 1, lottery = 0)}{P(Y = ham|Viagra = 1, lottery = 0)} = \frac{0.80}{0.20} = 4.0$$

Using a MAP decision rule we predict ham in the top two cases and spam in the bottom two. Given that the full posterior distribution is all there is to know about the domain in a statistical sense, these predictions are the best we can do: they are Bayes-optimal.

Example marginal likelihoods

Y	$P(\text{Viagra} = 1 \mid Y)$	$P(\text{Viagra} = 0 \mid Y)$
spam	0.40	0.60
ham	0.12	0.88

Y	$P(\text{lottery} = 1 \mid Y)$	$P(\text{lottery} = 0 \mid Y)$
spam	0.21	0.79
ham	0.13	0.87

Using marginal likelihoods

Using the marginal likelihoods from before, we can approximate the likelihood ratios (the previously calculated odds from the full posterior distribution are shown in brackets):

$$\frac{P(Viagra = 0|Y = spam)}{P(Viagra = 0|Y = ham)}\frac{P(lottory = 0|Y = spam)}{P(Lottery = 0|Y = ham)} = \frac{0.60}{0.88}\frac{0.79}{0.87} = 0.62 \qquad (0.45)$$

$$\frac{P(Viagra = 0|Y = spam)}{P(Viagra = 0|Y = ham)}\frac{P(lottory = 1|Y = spam)}{P(Lottery = 1|Y = ham)} = \frac{0.60}{0.88}\frac{0.21}{0.13} = 1.1 \qquad (1.9)$$

$$\frac{P(Viagra = 1|Y = spam)}{P(Viagra = 1|Y = ham)}\frac{P(lottory = 0|Y = spam)}{P(Lottery = 0|Y = ham)} = \frac{0.40}{0.12}\frac{0.79}{0.87} = 3.0 \qquad (4.0)$$

$$\frac{P(Viagra = 1|Y = spam)}{P(Viagra = 1|Y = ham)}\frac{P(lottory = 1|Y = spam)}{P(Lottery = 1|Y = ham)} = \frac{0.40}{0.12}\frac{0.21}{0.13} = 5.4 \qquad (0.67)$$

We see that, using a maximum likelihood decision rule, our very simple model arrives at the Bayes-optimal prediction in the first three cases, but not in the fourth ('Viagra' and 'lottery' both present), where the marginal likelihoods are actually very misleading.

Logistic Regression

a probabilistic linear classifier

Logistic Regression

- The aim is to do a binary classification, so we can assume that there are two classes $Y \in \{0,1\}$

$$P(Y = 0) + P(Y = 1) = 1$$

- If $P(Y = 1|x) > P(Y = 0|x)$ then we predict the data to belong to class 1 and if $P(Y = 1|x) < P(Y = 0|x)$ then we predict the data to belong to class 0

- Alternatively, we can look at the ratio of posteriors:

$$if \ \frac{P(Y=1|x)}{P(Y=0|x)} > 1 \quad \rightarrow \text{predict class 1}$$

$$if \ \frac{P(Y=1|x)}{P(Y=0|x)} < 1 \quad \rightarrow \text{predict class 0}$$

Logistic Regression

- We can write the *posterior odds*:

$$\frac{P(Y=1|x)}{P(Y=0|x)} = \frac{P(x|Y=1)}{P(x|Y=0)}\frac{P(Y=1)}{P(Y=0)}$$

- If we take the $log_e\ (ln)$ then:

$$ln\frac{P(Y=1|x)}{P(Y=0|x)} = ln\frac{P(x|Y=1)}{P(x|Y=0)} + ln\frac{P(Y=1)}{P(Y=0)}$$

Logistic Regression

- *Simplifying assumption:* We assume the log likelihood ratio is a linear function of x

$$ln\frac{P(x|Y=1)}{P(x|Y=0)} = x\beta$$

- $ln\frac{P(Y=1)}{P(Y=0)}$ is a fixed value, so we can define $\beta_0 = ln\frac{P(Y=1)}{P(Y=0)}$
- So, we have:

$$ln\frac{P(Y=1|x)}{P(Y=0|x)} = x\beta + \beta_0$$

Now we have a linear solution to our problem and this is what makes *Logistic Regression* a linear model.

Logistic Regression

- How to get our probability values:

$$ln\frac{P(Y=1|x)}{1-P(Y=1|x)} = x\beta + \beta_0$$

$$\frac{P(Y=1|x)}{1-P(Y=1|x)} = e^{x\beta+\beta_0}$$

$$P(Y=1|x) = \frac{e^{x\beta+\beta_0}}{1+e^{x\beta+\beta_0}} = \frac{1}{1+e^{-(x\beta+\beta_0)}}$$

Generalises to multiple class versions (Y can have more than two values).

Minimum Description Length Principle

Once again, the MAP hypothesis:

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

Which is equivalent to

$$h_{MAP} = \arg\max_{h \in H} \; [\log_2 P(D|h) + \log_2 P(h)]$$

Or

$$h_{MAP} = \arg\min_{h \in H} [-\log_2 P(D|h) - \log_2 P(h)]$$

Minimum Description Length Principle

Interestingly, this is an expression about a quantity of bits:

$$h_{MAP} = \arg\min_{h \in H} [-\log_2 P(D|h) - \log_2 P(h)] \qquad (1)$$

From information theory:

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

Minimum Description Length Principle

So interpret (1):

- $-\log_2 P(h)$ is length of h under optimal code. The length of your hypothesis
- $-\log_2 P(D|h)$ is length of D given h under optimal code. This is a notion of error/misclassification.

Note: assumes *optimal* encodings, when the priors and likelihoods are known. In practice, this is difficult, and makes a difference.

Minimum Description Length Principle

Occam's razor: a principle stating that, given all other things being equal, a shorter hypothesis for observed data should be favoured over a lengthier hypothesis.

MDL: prefer the hypothesis h that minimizes

$$h_{MAP} = \arg\min_{h \in H} [-\log_2 P(D|h) - \log_2 P(h)]$$

So from information theory perspective, the best hypothesis is the one that minimizes error and the size of hypothesis. So you want the simplest hypothesis that minimizes the error. So this is kind of a Bayesian argument for Occam's razor.

Summary

- We described the classification problem in machine learning
- We also outlined the issue of Inductive Bias
- Two major frameworks for classification were covered
    - Distance-based. The key ideas are geometric.
    - Probabilistic. The key ideas are Bayesian.
- We also discussed Logistic Regression
- So we have established the basis for learning classifiers
- Later we will see how to extend by building on these ideas

Acknowledgements

- Material derived from slides for the book
  "Elements of Statistical Learning (2nd Ed.)" by T. Hastie, R. Tibshirani & J. Friedman. Springer (2009) http://statweb.stanford.edu/~tibs/ElemStatLearn/
- Material derived from slides for the book
  "Machine Learning: A Probabilistic Perspective" by P. Murphy MIT Press (2012)
  http://www.cs.ubc.ca/~murphyk/MLbook
- Material derived from slides for the book "Machine Learning" by P. Flach Cambridge University Press (2012) http://cs.bris.ac.uk/~flach/mlbook
- Material derived from slides for the book
  "Bayesian Reasoning and Machine Learning" by D. Barber Cambridge University Press (2012)
  http://www.cs.ucl.ac.uk/staff/d.barber/brml
- Material derived from slides for the book "Machine Learning" by T. Mitchell McGraw-Hill (1997)
  http://www- 2.cs.cmu.edu/~tom/mlbook.html
- Material derived from slides for the course "Machine Learning" by A. Srinivasan BITS Pilani, Goa, India (2016)