Stat 406 Spring 2007: Homework 3

Out Wed 24 Jan, back Wed 31 Jan

1 Bivariate Gaussians

Let $X \sim \mathcal{N}(\mu, \Sigma)$ where $X \in \mathbb{R}^2$ and

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix} \tag{1}$$

where ρ is the correlation coefficient. Show that the pdf is given by

$$p(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left(\frac{(x_1-\mu_1)^2}{\sigma_1^2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2} - 2\rho\frac{(x_1-\mu_1)}{\sigma_1}\frac{(x_2-\mu_2)}{\sigma_2}\right)\right) \tag{2}$$

2 Uncorrelated does not imply independent unless *jointly* Gaussian

Let $X \sim \mathcal{N}(0,1)$ and $Y = WX$, where $p(W = -1) = p(W = 1) = 0.5$. It is clear that X and Y are not independent, since Y is a function of X.

1. Show $Y \sim \mathcal{N}(0,1)$. Thus X and Y are both Gaussian. Hint: To show the mean is zero, use the fact that X and W are independent. To show the varianec is 1, use the rule of iterated variance

$$\text{Var}\,(Y) = E\ \text{Var}\,(Y|W) + \text{Var}\,[E\,(Y|W)] \tag{3}$$

2. Show Cov$(X, Y) = 0$. Thus X and Y are uncorrelated but dependent, even though they are Gaussian. Hint: use the definition of covariance

$$\text{Cov}(X, Y) = E\,(XY) - E\,(X)E\,(Y) \tag{4}$$

and the rule of iterated expectation

$$E\,[XY] = E\,[E\,(XY|W)] \tag{5}$$

3 Likelihood ratio for Gaussians

Consider a binary classifier where the class conditional densities are MVN $p(x|y = j) = \mathcal{N}(x|\mu_j, \Sigma_j)$. By Bayes rule, we have

$$\log\frac{p(y=1|x)}{p(y=0|x)} = \log\frac{p(x|y=1)}{p(x|y=0)} + \log\frac{p(y=1)}{p(y=0)} \tag{6}$$

In other words, the log posterior ratio is the log likelihood ratio plus the log prior ratio. For each of the 4 cases in the table below, derive an expression for the log likelihood ratio $\log\frac{p(x|y=1)}{p(x|y=0)}$, simplifying as much as possible.

Form of Σ_j	Cov	Num parameters
Arbitrary	Σ_j	$Kd(d+1)/2$
Shared	$\Sigma_j = \Sigma$	$d(d+1)/2$
Shared, axis-aligned	$\Sigma_j = \Sigma$ with $\Sigma_{ij} = 0$ for $i \neq j$	d
Shared, spherical	$\Sigma_j = \sigma^2 I$	1

1	straight
2	magazines
3	issues
4	ray
5	enabled
6	head
7	improved
8	thread
9	libs
10	working

Figure 1: First 10 words in the vocabulary used for the NB exercise.

4 Maximum likelihood estimation of multinomials

Suppose $X \in \{1, 2\}$ and $Y \in \{1, 2, 3\}$. Define the joint distribution $P(X = j, Y = k) = \theta_{j,k}$. Consider the training data $\mathcal{D}$ below, where row i represents x_i and y_i:

X	Y
1	1
2	2
1	3
1	1
2	2
2	3

Find the maximum likelihood estimates

$$\hat{\theta}_{jk} = \arg\max \prod_{i=1}^{n} p(x_i, y_i|\theta) \tag{7}$$

Hint: just build the joint 2×3 table of counts and normalize so all numbers sum to one.

5 Naive Bayes classifier for document classification (Matlab)

[Be sure to download Data.zip, Code.zip and CodeEx.zip, and add these folders to your matlab path.]
Consider the problem of classifying email messages posted to online discussion boards into one of two classes, one for users of X Windows (class 1) and another for users of microsoft Windows (class 2). (This is analogous to **email spam filtering**.) There are 900 documents from each class; we divided them into training and text sets of equal size. To save space (and time), we ran word detection on the documents, and the data available to you consist of binary feature vectors for each document. Upon loading `Data/docdata.mat` the Matlab environment will contain variables `xtrain,xtest,ytrain,ytest`.
The identity of the 600 words is stored in the file 'Data/words.txt' you can load it into matlab by saying

```
vocab = textread('NB_words.txt','%s');
```

`vocab` is a cell array, so `vocab{t}` is the t'th word. You can print out the first 10 words using

```
for t=1:10
   fprintf(2,'%2d %20s\n', t,  vocab{t});
end
```

which produces the list in Figure 1.

1. Implement the following function

```
function theta = NBtrain(X,Y)
% Posterior mean estimate of Naive Bayes parameters
% Input:
% X(i,j) = 1 if word j appears in document i, otherwise X(i,j)=0
% Y(i) = class label of doc i (assumed to be 1 or 2)
% Output:
% theta(j,c) = probability of word j appearing in class c
```

which computes the following posterior mean estimate

$$\hat{\theta}_{jc} = \frac{N_{jc} + 1}{N_c + 2} \tag{8}$$

where N_{jc} counts the number of times word j appears in class c, N_c is the total number of documents in class c, and we have assumed a Beta(1,1) prior. Turn in your code.

2. Implement a function to classify each document, assuming uniform class priors $p(Y = 1) = p(Y = 2) = 0.5$.

```
function y = NBapply(X,theta)
% X(i,j) = 1 if word j appears in document i, otherwise X(i,j)=0
% theta(j,c) = prob of word j in class c
% y(i) = most probable class for X(i,:)
```

Here $y(i) = \arg\max_y p(Y = y|X(i,:))$ is the most probable class label for document i. Since $p(Y = y|\vec{x}) \propto p(\vec{x}|Y = y)$ is a small number, you will need to use logs to avoid underflow. (You don't necessarily need the logsumexp trick, because it suffices to compute the log likelihood $p(\vec{x}|y)$ rather than the normalized posterior $p(y|\vec{x})$, but you will need to use logs somehow!) Turn in your code.

3. Use `NBtrain` on the data in `xtrain,ytrain`. Compute the misclassification rates (i.e., the number of documents that you mis-classified) on the training set (by using `NBapply` on `xtrain,ytrain`) and on the test set (by using `NBapply` on `xtest,ytest`). Sanity check: You should get test error of **0.1867**.

4. The provided function (in CodeEx) `NBcv` computes the K-fold cross-validation error. (This calls your functions `NBtrain` and `NBapply`.) $K = 1$ means no cross-validation, that is error is simply computed on the whole training set. Use this to compute the 10-fold error rate on the training set. How does this compare to the (non cross validated) training and test error?

5. Plot (as histograms) the class-conditional densities $p(x_j = 1|y = c, \theta_c)$ for classes $c = 1, 2$ and words $j = 1 : 600$. You should get the same result as Figure 2.

6. What are the 5 most likely words in each class?

7. It is clear that the most probable words are not very discriminative. One way to measure how much information a word (feature) $X_j \in \{0, 1\}$ conveys about the class label $Y \in \{1, 2\}$ is by computing the **mutual information** between X_j and Y, denoted $I(X_j, Y)$, and defined as

$$mi(j) = I(X_j, Y) = \sum_{x=0}^{1} \sum_{c=1}^{2} p(X_j = x, Y = c) \log \frac{p(X_j = x, Y = c)}{p(X_j = x)p(Y = c)} \tag{9}$$

If we assume equal class priors, $p(Y = 1) = p(Y = 2) = 0.5$, then

$$p(X_j = 1, Y = c) = p(X_j|Y = y)p(Y = c) = \frac{\theta_{jc}}{2} \tag{10}$$

Use the provided function (jn CodeEx) `NBmi` to compute the 5 words with the highest mutual information with the class label. (Use the θ's estimated on `xtrain,ytrain`.) List the words along with the corresponding values of `MI`. (As a sanity check, the first word should be "windows" with an MI of 0.2150.)

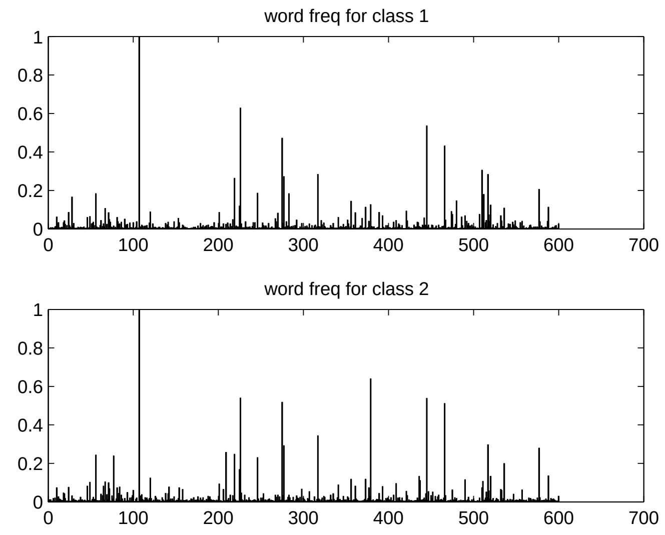

Figure 2: Class conditional densities $p(x_j = 1|c)$ for two document classes. The big spike at index 107 corresponds to the word “subject”, which occurs in both classes with probability 1.