CSC412 – Assignment #2 Solutions

March 7, 2002

1 Gamma Distribution (3+3+3+4 marks)

The *gamma* distribution is a distribution on positive real values $0 < x < \infty$. It has the form:

$$p(x) = \frac{1}{sG(c)} \left(\frac{x}{s}\right)^{c-1} \exp\left\{-\frac{x}{s}\right\}$$

$G(c)$ is a function that makes the distribution normalize properly; $s, c > 0$ are positive scale and shape parameters.

- Write this distribution in the exponential family form.

  $$p(x) = \exp\left(-\frac{x}{s} + (c-1)\log x - c\log s - \log G(c)\right)$$

  where the natural parameters are $[-1/s \;\; c-1]$ and the sufficient statistics are $[x \;\; \log x]$, and $\exp(c\log s + \log G(c))$ is the normalizing constant.

- What are the sufficient statistics of a dataset $\mathcal{D} = \{x^1, x^2, \ldots, x^M\}$ for the gamma distribution?

  The sufficient statistics are $\sum_{m=1}^{M} x^m$ and $\sum_{m=1}^{M} \log x^m$.

- Write down the log-liklihood function $\ell(s, c; \mathcal{D}) = \log p(x^1, x^2, \ldots, x^M | s, c)$ in terms of the parameters $(s; c)$, the sufficient statistics of $\mathcal{D}$, and the function $G(c)$.

  $$\ell(s, c; \mathcal{D}) = -\frac{1}{s}\sum_{m=1}^{M} x^m + (c-1)\sum_{m=1}^{M} \log x^m - c\log s - \log G(c)$$

- For a fixed parameter s^*, find the maximum likelihood estimate c^* in terms of the sufficient statistics and s^*.

  Differentiate $\ell(s, c; \mathcal{D})$ with respect to s, we get

  $$\frac{\partial \ell(s, c; \mathcal{D})}{\partial s} = \frac{1}{s^2}\sum_{m=1}^{M} x^m - \frac{c}{s} = 0$$
  $$c^* = \frac{1}{s}\sum_{m=1}^{M} x^m$$

2 Flipping Coins (7 marks)

- Write a MATLAB program or script that simulates measuring a biased coin which is flipped M times.

```
function mltheta = flip(theta,M,N);

% Returns the ML estimate of the bias of a coin based on a random sample of
% M coin flips, where the bias is theta.
```

```
% This is repeated N times, and the individual estimates are stored in
% the vector mltheta.
% Also plots a histogram of mltheta.

mltheta = mean(rand(N,M)<theta,2);
hist(mltheta,100);
```

3 Class-Conditional Gaussians (5+5+5+5 marks)

In this question, you'll derive for yourself the maximum likelihood estimates for class-conditional Gaussians with independent features (spherical covariance matrices). Start with the following generative model for a discrete class label $y \in (1, 2, \ldots, K)$ and a real valued vector of N features $\mathbf{x} = (x_1, x_2, \ldots, x_N)$:

$$p(y = k) = \alpha_k$$

$$p(\mathbf{x}|y = k) = (2\pi\sigma^2)^{-N/2} \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n - \mu_{kn})^2\right\}$$

where α_k is the prior on class k, σ^2 is the shared variance for all features in all classes, and μ_{kn} is the mean of the feature n conditioned on class k.

- Use Bayes' rule to invert the model above and write the expression for $p(y = k|\mathbf{x})$.

$$p(y = k|\mathbf{x}) = \frac{\alpha_k \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n - \mu_{kn})^2\right\}}{\sum_k \alpha_k \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n - \mu_{kn})^2\right\}}$$

- Write down the expression for the likelihood function $\ell(\theta; \mathcal{D}) = \log p(y^1, x^1, y^2, x^2, \ldots, y^M, x^M|\theta)$ of a particular dataset $\mathcal{D} = \{y^1, x^1, y^2, x^2, \ldots, y^M, x^M\}$ with parameters $\theta = \{\alpha, \mu, \sigma^2\}$.

$$\begin{aligned}
l(\theta; \mathcal{D}) &= \sum_{m=1}^{M}[y^m = k]\left(\sum_k \log\alpha_k - \frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n^m - \mu_{kn})^2\right) \\
&= \sum_k \log\alpha_k \sum_{m=1}^{M}[y^m = k] - \frac{NM}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_k\sum_{n=1}^{N}\sum_{m=1}^{M}[y^m = k](x_n^m - \mu_{kn})^2
\end{aligned}$$

  - It is important to write the expression for $p(x, y)$ in such a way that the log likelihood is linear in the indicator variables (i.e. y). This is so that when it comes time to differentiate, the expression remains simple. In this case I use a product over the possible values k of y raised to the power of $[y = k]$:

$$p(x, y) = \prod_k \left(\alpha_k \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n - \mu_{kn})^2\right\}\right)^{[y=k]}$$

- Take partial derivatives of the likelihood with respect to each of the parameters μ_{nk} and with respect to the shared variance σ^2. (Don't worry about α.)

- Set these partial derivatives to zero and solve for the optimal (maximum likelihood) parameter values μ_{nk} and σ^2 for classification.

$$\frac{\partial \ell(\theta;\mathcal{D})}{\partial \mu_{kn}} = \frac{1}{\sigma^2}\sum_{m=1}^{M}[y^m = k](\mu_{kn} - x_n^m) = 0$$

$$\mu_{kn} = \frac{\sum_{m=1}^{M}[y^m = k]x_n^m}{\sum_{m=1}^{M}[y^m = k]}$$

$$\frac{\partial \ell(\theta;\mathcal{D})}{\partial \sigma^2} = -\frac{NM}{2\sigma^2} + \frac{1}{2\sigma^4}\sum_k \sum_{n=1}^{N}\sum_{m=1}^{M}[y^m = k](x_n^m - \mu_{kn})^2 = 0$$

$$\sigma^2 = \frac{1}{NM}\sum_k \sum_{n=1}^{N}\sum_{m=1}^{M}[y^m = k](x_n^m - \mu_{kn})^2$$

4 Handwritten Digit Classification (8+12+15 marks)

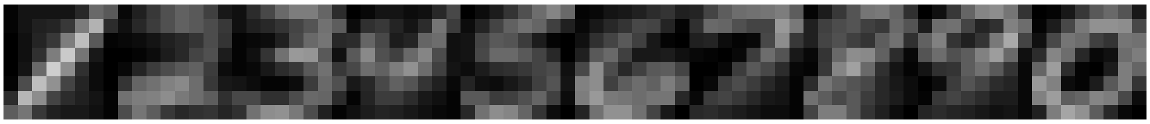
A sample solution for the question is as follows:

```
load a2digits

numpixel = 64;  onepixel = ones(1,numpixel);
numtrain = 700; onetrain = ones(1,numtrain);
numtest  = 400; onetest  = ones(1,numtest );
numclass = 10;  oneclass = ones(1,numclass);


% conditional Gaussian fitting
cg_means = mean(digits_train,2);
delta = digits_train-cg_means(:,onetrain,:);
cg_var = mean(delta(:).^2);
cg_std = sqrt(cg_var);

% naive Bayes fitting
nb_means = mean(digits_train>.5,2);

% draw means
figure(1); clf; colormap gray;
subplot(211);
imagesc(reshape(permute(reshape(cg_means,...
        [8 8 10]),[2 1 3]),[8 80]),[0 1]);
axis equal; axis off;
title(['Conditional Gaussian Means, variance=' num2str(cg_var) ...
       ', standard deviation=' num2str(cg_std)]);
subplot(212);
imagesc(reshape(permute(reshape(nb_means,[8 8 10]),[2 1 3]),[8 80]),[0 1]);
axis equal; axis off;
title('Naive Bayes Means');

% 3 by 7 plot
figure(2); clf; colormap gray
cases = [3 14 242 112 364 319 72];
classes = [2 3 5 6 7 9 10];
for ii=1:length(cases),
  digit = digits_test(:,cases(ii),classes(ii));

  % conditional Gaussian posterior
```

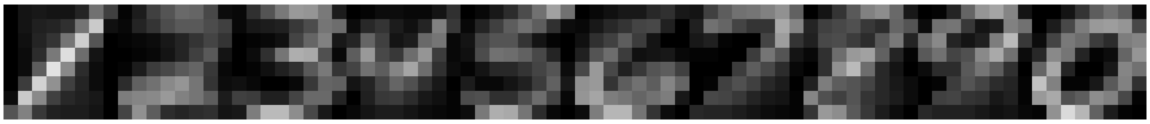
```
  delta = digit(:,oneclass)-squeeze(cg_means);
  tmp = -.5/cg_var*sum(delta.^2,1);
  cg_post = exp(tmp - max(tmp));
  cg_post = cg_post/sum(cg_post);

  % naive Bayes posterior
  tmp = (digit>.5)'*log(squeeze(nb_means))+...
        (digit< .5)'*log(squeeze(1-nb_means));
  nb_post = exp(tmp-max(tmp));
  nb_post = nb_post/sum(nb_post);

  % Draw outputs
  subplot(3,7,ii);
  imagesc(reshape(digit,[8 8])',[0 1]);
  axis equal; axis off;
  title(['digit ' num2str(rem(classes(ii),10)) ...
         ', case ' num2str(cases(ii))]);

  subplot(3,7,ii+7);
  bar(cg_post);
  axis([.5 10.5 0 1]);
  if ii==1, ylabel('Gaussian'); end;

  subplot(3,7,ii+14);
  bar(nb_post);
  axis([.5 10.5 0 1]);
  if ii==1, ylabel('Naive Bayes'); end;
end;

% Error outputs
digits = digits_test(:,:,:,oneclass);
correct_class = repmat(1:10,[numtest 1]);

% conditional Gaussians
means = permute(cg_means(:,onetest,:,oneclass),[1 2 4 3]);
delta = digits - means;
tmp = squeeze(-sum(delta.^2,1));
[tmp predicted_class] = max(tmp,[],3);
cg_numerror = sum(predicted_class~=correct_class,1);

% naive Bayes
digits = digits>.5;
means = permute(nb_means(:,onetest,:,oneclass),[1 2 4 3]);
tmp = squeeze(sum(log(digits.*means+(1-digits).*(1-means)),1));
[tmp predicted_class] = max(tmp,[],3);
nb_numerror = sum(predicted_class~=correct_class,1);

% display errors
fprintf(1,' Class  | Gaussian errors | Naive Bayes errors\n');
fprintf(1,'  %3d   |       %3d       |       %3d\n',...
        [1:9 0; cg_numerror; nb_numerror]);
fprintf(1,'overall |      %2.2f%%     |      %2.2f%%\n',...
        mean(cg_numerror)/4,mean(nb_numerror)/4);
```

The means obtained are

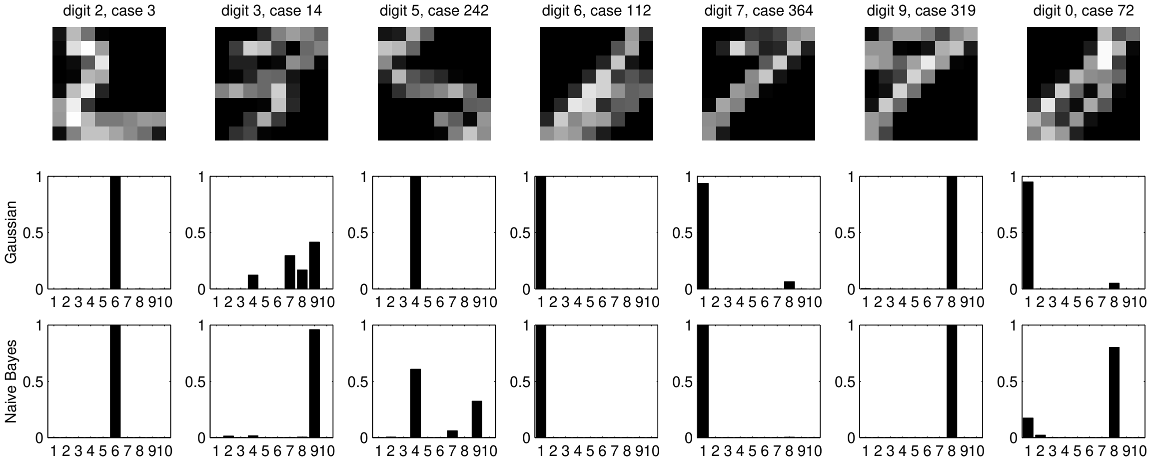
Conditional Gaussian Means, variance=0.063351, standard deviation=0.2517



Naive Bayes Means



The posteriors for the seven test cases are



and the errors are

Class	Gaussian errors	Naive Bayes errors
1	69	87
2	81	104
3	63	91
4	61	85
5	68	111
6	44	60
7	63	89
8	109	121
9	110	133
0	53	58
overall	18.02%	23.48%

- When drawing the digits with `imagesc`, you should set `axis equal` so that they do not appear skewed.
- Please draw the posteriors (rather than log posteriors or negative log posteriors).
- Remember to renormalize the posteriors (although for testing purposes it is not necessary).
- When calculating the posteriors use the softmax trick to avoid numerical over- or under-flow. In particular, first calculate $l_k = \log p(x, y = k)$ for each value of k, then the posteriors are:

$$p(y = k|x) = \frac{\exp(l_k - \max_{k'}(l_{k'}))}{\sum_k \exp(l_k - \max_{k'}(l_{k'}))}$$

  Notice that the maximum value of $\exp(l_k - \max_{k'}(l_{k'}))$ is exactly 1.