Engineering Tripos Part IIB FOURTH YEAR

Module 4F10: STATISTICAL PATTERN RECOGNITION

Solutions to Examples Paper 1

1. Average risk in choosing class ω_i is

$$\begin{aligned} R(\omega_i|\boldsymbol{x}) &= \sum_{j=1}^{c} \lambda(\omega_i|\omega_j) P(\omega_j|\boldsymbol{x}) \\ &= 0.P(\omega_i|\boldsymbol{x}) + \sum_{j=1, j\neq i}^{c} \lambda_s P(\omega_j|\boldsymbol{x}) \end{aligned}$$

where $\lambda(\omega_i|\omega_j)$ is used to mean the cost of choosing class ω_i where the true class is ω_j.

Hence

$$R(\omega_i|\boldsymbol{x}) = \lambda_s \left(1 - P(\omega_i|\boldsymbol{x})\right)$$

Associate $\boldsymbol{x}$ with class ω_i if highest posterior class probability and the average risk is less than the cost of rejection

$$\begin{aligned} \lambda_s \left(1 - P(\omega_i|\boldsymbol{x})\right) &\leq \lambda_r \\ P(\omega_i|\boldsymbol{x}) &\geq 1 - \lambda_r/\lambda_s \end{aligned}$$

If the ratio λ_r/λ_s is close to 1 then the reject region will tend to zero. If λ_r/λ_s is close to zero then nearly all examples will be rejected.

2. The computational cost for the 3 systems are

   (a) Diagonal covariance the cost is $2d$ multiply accumulates.

   (b) Full covariance the cost is $d^2 + d$ multiply accumulates.

   (c) M component diagonal system the cost is $2Md$.

   In all cases the inverse covariance matrix is stored and the portion of the Gaussian PDF not dependent on the observations is stored as a constant.

   As a practical matter for the Gaussian mixture case, $\log(P(\omega_m))$ is added to the constant in advance and the computation can be arranged to use in the sum (in the log domain)

$$\log(\exp(l_i(\mathbf{x})) + \exp(l_j(\mathbf{x}))) = l_i(\mathbf{x}) + \log(1 + \exp(l_j(\mathbf{x}) - l_i(\mathbf{x})))$$

assuming that $l_i(\mathbf{x}) \geq l_j(\mathbf{x})$ and

$$l_i(\mathbf{x}) = \log(P(\omega_i)) + \log(\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)))$$

This often saves one exponential calculation, the exponential is only calculated when the value of the difference in logs is above a threshold, and handles any dynamic range issues.

Note an approximation that is sometimes used for a GMM is that the overall likelihood approximated by using only the largest of the component log-likelihoods i.e.

$$\log(p(\mathbf{x})) \approx \max_m(\log(P(\omega_m)) + \log(\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)))$$

This saves adding the logs above but is not useful for training where more precise calculation is required.

3. (a) In this case, the variance is equal to 1 for each of the single dimensional Gaussians. The log likelihood of the model for single dimensional Gaussians and two mixture components is

$$l(\boldsymbol{\theta}) = \sum_{k=1}^{n} \ln \left[\sum_{m=1}^{2} c_m \frac{1}{(2\pi)^{1/2}} \exp \left\{ \frac{-(x_k - \mu_m)^2}{2} \right\} \right]$$

Here $n = 9$ and the question asks for the likelihood with $c_1 = c_2 = 0.5$ which can be calculated as

$$\prod_{k=1}^{9} \sum_{m=1}^{2} \frac{1}{q} \exp \left\{ \frac{-(x_k - \mu_m)^2}{2} \right\}$$

where $1/q = 0.199$. Substituting in the data values and the mean values yields the total likelihood of the data as 2.262×10^{-7}.

(b) To compute the re-estimated means and component priors / mixture weights, find the posterior probabilities of each mixture component for each data sample and accumulate numerator and denominator statistics. This is most easily done by writing a small program/script. The posterior probabilities of each mixture component is given in the table below.

x	$P(\text{comp1}\|x)$	$P(\text{comp1}\|x)$
-1.5	0.9933	0.0067
-0.5	0.9526	0.0474
0.1	0.8581	0.1419
0.3	0.8022	0.1978
0.9	0.5498	0.4502
1.3	0.3543	0.6457
1.9	0.1419	0.8581
2.3	0.0691	0.9309
3.0	0.0180	0.9820

Then for each mixture component accumulate for mean re-estimation the numerator $\sum_{k=1}^{9} x_k P(m|x_k)$ and for the denominator $\sum_{k=1}^{9} P(m|x_k)$. The same statistics used for the denominator are also needed for component prior re-estimation.

Computing these values leads to $\hat{\mu}_1 = -0.0426$; $\hat{\mu}_2 = 1.878$; $\hat{c}_1 = 0.5266$; $\hat{c}_2 = 0.4734$. Use of these values to re-compute the likelihood yields an increase over the initial values.

4. From lecture notes we can write the auxiliary function as

$$\mathcal{Q}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) = \sum_{m=1}^{M} \sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i) \sum_{k=1}^{d} [x_{ik} \log(\lambda_{mk}) + (1 - x_{ik}) \log(1 - \lambda_{mk})]$$

Differentiate this with respect to λ_{qr} give

$$\frac{\partial \mathcal{Q}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})}{\partial \lambda_{qr}} = \sum_{i=1}^{n} P(\omega_q|\mathbf{x}_i) \left[\frac{x_{ir}}{\lambda_{qr}} - \frac{(1 - x_{ir})}{(1 - \lambda_{qr})} \right]$$

Equating to zero gives

$$(1 - \lambda_{qr}) \sum_{i=1}^{n} P(\omega_q|\mathbf{x}_i) x_{ir} = \lambda_{qr} \sum_{i=1}^{n} P(\omega_q|\mathbf{x}_i)(1 - x_{ir})$$

Rearranging yields the answer.

5. We can write

$$x_i = t_i + z$$

where z is Gaussian distributed, mean 0 and variance 1.

(a) Since the two are independent and both Gaussian distributed we know that

$$p(x_i|\theta) = \mathcal{N}(x_i; \mu, \sigma^2 + 1)$$

We can write the log-likelihood of the training data as

$$\begin{aligned} l(\boldsymbol{\theta}) &= \sum_{i=1}^{n} \log(\mathcal{N}(x_i, \mu, 1 + \sigma^2) \\ &= \sum_{i=1}^{n} -\frac{1}{2} \left(\log(2\pi(1 + \sigma^2)) + \frac{(x_i - \mu)^2}{1 + \sigma^2} \right) \end{aligned}$$

Differentiating

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \mu} = \sum_{i=1}^{n} \frac{x_i - \mu}{1 + \sigma^2}$$

Equating to zero gives

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

For the variance

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \sigma^2} = \sum_{i=1}^{n} -\frac{1}{2}\left(\frac{1}{(1+\sigma^2)} - \frac{(x_i-\mu)^2}{(1+\sigma^2)^2}\right)$$

Equating to zero and using the ML estimate for μ

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

(b) This is the same set up as described in lecture. Let z_i be the noise associated with observation i. So

$$p(x_i|z_i,\theta) = \mathcal{N}(x_i; \mu + z_i, \sigma^2)$$

We first need to compute the posterior $p(z_i|x_i,\theta)$

$$\begin{aligned} p(z_i|x_i,\theta) &= \frac{p(x_i|z_i,\theta)p(z_i)}{p(x_i|\theta)} \\ &= \mathcal{N}\left(z_i; \frac{(x_i-\mu)}{(1+\sigma^2)}, \frac{\sigma^2}{(1+\sigma^2)}\right) \end{aligned}$$

So writing down the auxiliary function

$$\begin{aligned} \mathcal{Q}(\theta,\hat{\theta}) &= \sum_{i=1}^{n}\int (p(z_i|x_i,\theta)\log(p(x_i,z_i|\hat{\theta}))dz_i \\ &= \sum_{i=1}^{n}\int (p(z_i|x_i,\theta)\log(p(x_i|z_i,\hat{\theta}))dz_i \\ &\quad + \sum_{i=1}^{n}\int (p(z_i|x_i,\theta)\log(p(z_i))dz_i \end{aligned}$$

The second term is not dependent on the new model parameters, the distribution of z_i is known. This leaves the first term. From the previous definitions

$$\begin{aligned} \tilde{\mathcal{Q}}(\theta,\hat{\theta}) &= \sum_{i=1}^{n}\int p(z_i|x_i,\theta)\log(p(x_i|z_i,\hat{\theta}))dz_i \\ &= \sum_{i=1}^{n}\int p(z_i|x_i,\theta)\left[\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{(x_i-z_i-\hat{\mu})^2}{2\sigma^2}\right]dz_i \\ &= \sum_{i=1}^{n}\left[\log\left(\frac{1}{\sqrt{2\pi\hat{\sigma}^2}}\right) - \frac{(x_i-\hat{\mu})^2 - 2(x_i-\hat{\mu})\mathcal{E}\{z_i|\theta,x_i\} + \mathcal{E}\{z_i^2|\theta,x_i\}}{2\hat{\sigma}^2}\right] \end{aligned}$$

We know that

$$\begin{aligned}\mathcal{E}\{z_i|\theta, x_i\} &= \frac{(x_i-\mu)}{(1+\sigma^2)} \\ \mathcal{E}\{z_i^2|\theta, x_i\} &= \frac{\sigma^2}{(1+\sigma^2)} + \left(\frac{(x_i-\mu)}{(1+\sigma^2)}\right)^2\end{aligned}$$

Differentiating with respect to $\hat{\mu}$ gives

$$\frac{\partial \tilde{\mathcal{Q}}(\theta, \hat{\theta})}{\partial \hat{\mu}} = \sum_{i=1}^{n} \frac{1}{\hat{\sigma}^2}\left(x_i - \hat{\mu} - \mathcal{E}\{z_i|\theta, x_i\}\right)$$

so

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

Differentiating with respect to σ^2

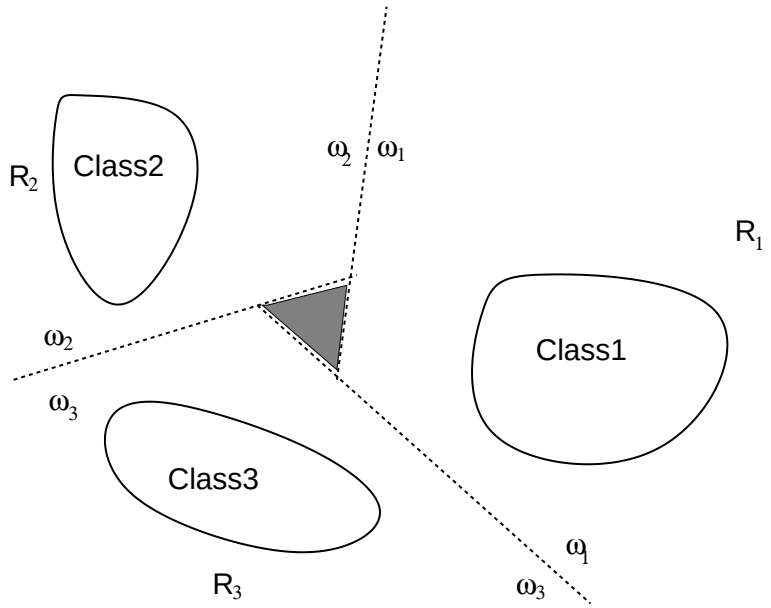
$$\frac{\partial \tilde{\mathcal{Q}}(\theta, \hat{\theta})}{\partial \hat{\sigma}^2} = \frac{1}{2}\sum_{i=1}^{n}\left[\frac{-1}{\hat{\sigma}^2} + \frac{(x_i-\hat{\mu})^2 - 2(x_i-\hat{\mu})\mathcal{E}\{z_i|\theta, x_i\} + \mathcal{E}\{z_i^2|\theta, x_i\}}{(\hat{\sigma}^2)^2}\right]$$

Equating to zero gives

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}\left[(x_i-\hat{\mu})^2 - 2(x_i-\hat{\mu})\mathcal{E}\{z_i|\theta, x_i\} + \mathcal{E}\{z_i^2|\theta, x_i\}\right]$$

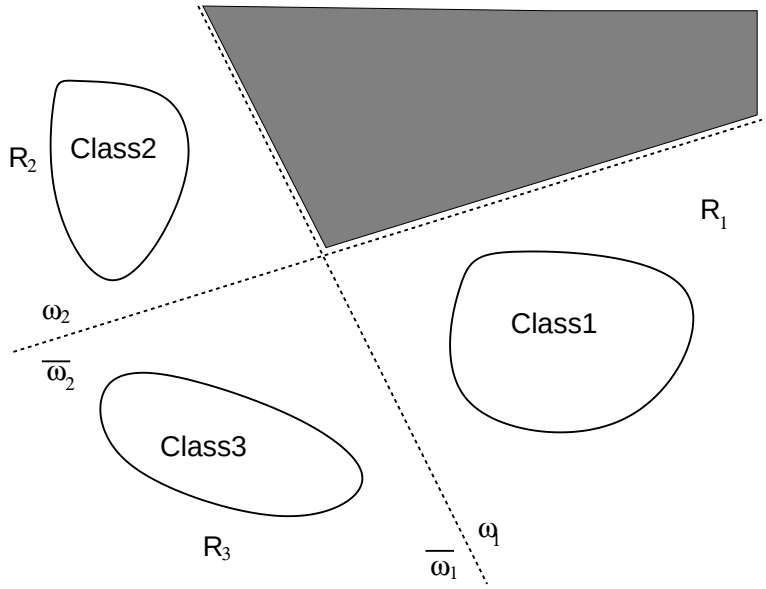
This problem is simple to solve using standard optimisation techniques. For EM the correct answer is eventually obtained, but only after many iterations. For situations where direct optimisation is not simple, EM is useful, for example mixture models. Gradient descent could be used in these situations, but do not have the guaranteed stability of EM.

6. For the one v one classifiers: it is necessary to train and decode with $K(K-1)/2$ classifiers. The no decision region is shown below



The region marked in gray is "no decision". In this case it appears to be both class 1 and class 2

For the one v rest classifiers: it is necessary to train and decode with $(K-1)$ classifiers. The no decision region is shown below



7. Total number of weights in the system is

   - input to hidden layer: $(d+1)M$
   - the $L-1$ hidden to hidden: $(L-1)M(M+1)$

- hidden to output $(M+1)K$

The number of hidden layers determines the decision boundaries that can be produced (see lecture notes), the activation function determines the nature of the output - binary (step), sum to one (soft max), continuous (linear) etc. The number of hidden units should be large enough to model the problem, but small enough so that *generalisation* is not an issue.

8.

$$\phi(z) = \frac{1}{1+\exp(-z)}, \quad \begin{aligned} \frac{\partial \phi(z)}{\partial z} &= \frac{\exp(-z)}{(1+\exp(-z))^2} \\ &= \phi(z)(1-\phi(z)) \end{aligned}$$

The activation function affects the form of the error back propagation algorithm. The derivation given in lectures assumes a sigmoid, however the output layer can be more complex if a sum squared error is used with a softmax function (not not if used with a cross-entropy measure) since in this case the partial derivative for a particular weight in the output layer depends on all output values due to the normalisation in the softmax.

9. If the gradient is approximately constant then we can write

$$\boldsymbol{\Delta}\mathbf{w}[\tau] = -\eta \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} + \alpha \boldsymbol{\Delta}\mathbf{w}[\tau-1]$$

Substituting back yields

$$\begin{aligned} \boldsymbol{\Delta}\mathbf{w}[\tau] &= -\eta \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} + \alpha \left(-\eta \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} + (....)\right) \\ &= -\eta \left(1+\alpha+\alpha^2+...\right) \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} \end{aligned}$$

If $\alpha < 1$ then the sum of the infinite GP give

$$\boldsymbol{\Delta}\mathbf{w}[\tau] = -\left(\frac{\eta}{1-\alpha}\right) \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]}$$

If the solution is oscillating then we can write (approximately)

$$\begin{aligned} \boldsymbol{\Delta}\mathbf{w}[\tau] &= -\eta \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} + \alpha \left(+\eta \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} + (....)\right) \\ &= -\eta \left(1-\alpha+\alpha^2-...\right) \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} \\ &= -\eta \left((1-\alpha)(1+\alpha^2+\alpha^4...)\right) \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} \\ &= -\eta \frac{(1-\alpha)}{(1-\alpha^2)} \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} \\ &= -\left(\frac{\eta}{1+\alpha}\right) \left.\boldsymbol{\nabla} E(\mathbf{w})\right|_{\mathbf{w}[0]} \end{aligned}$$

10. (a) The Hessian may be used to obtain the Newton direction. This requires computing $\mathbf{H}^{-1}\mathbf{g}$. (see lecture notes for more details).

(b)

$$\frac{\partial E}{\partial w_{ij}} = \sum_{p=1}^{n}(y(x_p) - t(x_p))\frac{\partial y(x_p)}{\partial w_{ij}}$$

and

$$\frac{\partial^2 E}{\partial w_{ij}\partial w_{lk}} = \sum_{p=1}^{n}\frac{\partial y(x_p)}{\partial w_{lk}}\frac{\partial y(x_p)}{\partial w_{ij}} + \sum_{p=1}^{n}(y(x_p) - t(x_p))\frac{\partial^2 y(x_p)}{\partial w_{ij}\partial w_{lk}}$$

For the conditions described the network will train so that

$$y(x_p) = t(x_p)$$

In this condition the second term is zero.

(c) From the conditions given

$$\mathbf{H}_{N+1} = \mathbf{H}_N + \mathbf{g}^{(N+1)}(\mathbf{g}^{(N+1)})'$$

Consider the inverse

$$\begin{aligned}
\mathbf{H}_{N+1}^{-1} &= \left(\mathbf{H}_N + \mathbf{g}^{(N+1)}(\mathbf{g}^{(N+1)})'\right)^{-1} \\
&= \mathbf{H}_N^{-1} - \mathbf{H}_N^{-1}\mathbf{g}^{(N+1)}\left(1 + \mathbf{g}^{(N+1)\prime}\mathbf{H}_N^{-1}(\mathbf{g}^{(N+1)})\right)^{-1}(\mathbf{g}^{(N+1)})'\mathbf{H}_N^{-1} \\
&= \mathbf{H}_N^{-1} - \frac{\mathbf{H}_N^{-1}\mathbf{g}^{(N+1)}(\mathbf{g}^{(N+1)})'\mathbf{H}_N^{-1}}{1 + \mathbf{g}^{(N+1)\prime}\mathbf{H}_N^{-1}(\mathbf{g}^{(N+1)})}
\end{aligned}$$

The calculation of the inverse can be computationally expensive for large numbers of weights (naive implementation $\mathcal{O}(W^3)$). This scheme directly calculates the inverse. An initial value is needed for this scheme ($\mathbf{H}_0$). The simplest approach is to use a diagonal matrix with very small values on the leading diagonal (easy to invert and will not distort the final results).

M.J.F. Gales
P.C. Woodland
October 2003 - February 2007