2007 IIB 4F10 Dr M Gales
mjfg@eng.cam.ac.uk

Solutions to 4F10 Pattern Processing, 2007

1. *Gaussian classifiers and Fisher discriminant*

(a) The minimum error rate classifier simply assigns the test data sample to the class with the highest posterior probability. For each class i this is computed using Bayes' Theorem

$$P(w_i|\mathbf{x}) = \frac{P_i p(\mathbf{x}|\omega_i)}{\sum_{j=1}^{2} P_j p(\mathbf{x}|\omega_j)}$$

Since the denominator is independent of the class find (taking logs of the above and substituting for a Gaussian) [15%]

$$\arg\max_i \ln P_i - 1/2 \ln |\boldsymbol{\Sigma}_i| - 1/2(\mathbf{x}-\boldsymbol{\mu}_i)^{'}\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)$$

(b) The decision boundary is values of $\mathbf{x}$ when $P(\omega_1|\mathbf{x}) = P(\omega_2|\mathbf{x})$. So in this case defined by values of $\mathbf{x}$ such that

$$\ln P_1 - 1/2 \ln |\boldsymbol{\Sigma}_1| - 1/2(x-\boldsymbol{\mu}_1)^{'}\boldsymbol{\Sigma}_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1) = \ln P_2 - 1/2 \ln |\boldsymbol{\Sigma}_2| - 1/2(\mathbf{x}-\boldsymbol{\mu}_2)^{'}\boldsymbol{\Sigma}_2^{-1}(\mathbf{x}-\boldsymbol{\mu}_2)$$

or re-arranging so that

$$\begin{aligned}(\boldsymbol{x}-\boldsymbol{\mu}_1)'\boldsymbol{\Sigma}_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) - (\boldsymbol{x}-\boldsymbol{\mu}_2)'\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) \\ - \ln\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} - 2\ln\frac{P_1}{P_2} = 0\end{aligned}$$

Hence

$$\begin{aligned}&\mathbf{x}'(\boldsymbol{\Sigma}_1^{-1} - \boldsymbol{\Sigma}_2^{-1})\boldsymbol{x} + 2(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu_2} - \boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu_1})^{'}\boldsymbol{x} \\ &+\boldsymbol{\mu}_1'\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2'\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2 - \ln\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} - 2\ln\frac{P_1}{P_2} \quad = 0\end{aligned}$$

which is the form as required and the terms $\mathbf{A}$, $\mathbf{b}$ and c can be clearly identified. [25%]

(c)(i) The Fisher discriminant $\mathbf{w}$ is found by maximising the ratio of the projected means (in the direction of the Fisher discriminant) to the projected average within-class scatter matrix.

$$E(\mathbf{w}) = -\frac{(\overline{\mu}_1 - \overline{\mu}_2)^2}{\overline{s}_1 + \overline{s}_2}$$

where $\overline{s}_j$ and $\overline{\mu}_j$ are the projected scatter matrix and mean for class ω_j

The discriminant is in the direction (note that lectures discuss this in terms of scatter matrices from data so this is a slightly different presentation) [15%]

$$\mathbf{w} = [1/2(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2)]^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

(c)(ii) The Fisher discriminant gives a direction but an offset for the decision boundary. The discrimination rule is set takes the Fisher direction $\mathbf{w}$ and projects data onto this. This value is then compared with a threshold T to form the decision choosing either class ω_1 or class ω_2

Hence

$$\mathbf{w}'\mathbf{x} \underset{\omega_2}{\overset{\omega_1}{\lessgtr}} T$$

The value of T would be set to minimise error, increase class separation etc. [15%]

(c)(iii)

$$\begin{aligned} \mathbf{w} &= \begin{bmatrix} 2/5 & 0 \\ 0 & 2/5 \end{bmatrix} \begin{bmatrix} 2 \\ -2 \end{bmatrix} \\ &= \begin{bmatrix} 4/5 \\ -4/5 \end{bmatrix} \end{aligned}$$

An appropriate threshold in this case is zero (i.e. the decision boundary should pass through the origin at an angle of $\pi/4$) for minimum error. [15%]

(c)(iv) Clearly there must be no effective quadratic term in the Gaussian classifier (e.g. equal covariance matrices among the classes) and the offset must be set to be equal. However there are other cases of linear boundaries in Gaussian classifiers (even with different covariances) and the example in (c) (ii) is one such due to the problem symmetry. For common covariance matrices the $\mathbf{w}$ vector is in the Fisher direction. The discrimination rule offset is arbitary but in general would be set to minimise error, and hence in this case it would also be the minimum error decision boundary for the Gaussian classifier. [15%]

2. *Gaussian mixture models and EM*

(a) For a Gaussian mixture we have

$$p(\boldsymbol{x}) = \sum_{m=1}^{M} P(\omega_m)\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$$

The log-likelihood function for the mixture model is [10%]

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{n} \ln p(\boldsymbol{x}_i) = \sum_{i=1}^{n} \ln \left[\sum_{m=1}^{M} p(\boldsymbol{x}_i|m) P(\omega_m) \right]$$

(b) The underlying issue here is that the association of a particular mixture component value with a training vector is not known i.e. the association is through latent variables. If the log-likelihood function is differentiated and equated to zero, the expression results in a coupled set of non-linear equations, since the right-hand side of an estimation formula depends on the posterior probabilities of mixture component occupation which in turn depends on the model parameters. [10%]

(c) If we change the parameters from $\boldsymbol{\theta}$ to $\hat{\boldsymbol{\theta}}$, then the increase in log likelihood is

$$l(\hat{\boldsymbol{\theta}}) - l(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \left(\frac{p(\mathbf{x}_i|\hat{\boldsymbol{\theta}})}{p(\mathbf{x}_i|\boldsymbol{\theta})} \right)$$

For a mixture distribution, denoting the m^{th} mixture component as ω_m,

$$\begin{aligned} l(\hat{\boldsymbol{\theta}}) - l(\boldsymbol{\theta}) &= \sum_{i=1}^{n} \log \left(\frac{1}{p(\mathbf{x}_i|\boldsymbol{\theta})} \sum_{m=1}^{M} p(\mathbf{x}_i, \omega_m|\hat{\boldsymbol{\theta}}) \right) \\ &= \sum_{i=1}^{n} \log \left(\frac{1}{p(\mathbf{x}_i|\boldsymbol{\theta})} \sum_{m=1}^{M} \left(\frac{p(\mathbf{x}_i, \omega_m|\hat{\boldsymbol{\theta}}) P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})}{P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})} \right) \right) \end{aligned}$$

Since log() is strictly concave we can use Jensen's Inequality which states that for $\lambda_m \geq 0$ and $\sum_m \lambda_m = 1$

$$\log \left(\sum_{m=1}^{M} \lambda_m x_m \right) \geq \sum_{m=1}^{M} \lambda_m \log (x_m)$$

Now using the numerator $P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})$ as λ_m gives

$$l(\hat{\boldsymbol{\theta}}) - l(\boldsymbol{\theta}) \geq \sum_{i=1}^{n} \sum_{m=1}^{M} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta}) \log \left(\frac{p(\mathbf{x}_i, \omega_m|\hat{\boldsymbol{\theta}})}{p(\mathbf{x}_i|\boldsymbol{\theta}) P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})} \right)$$

which can be written as

$$l(\hat{\boldsymbol{\theta}}) - l(\boldsymbol{\theta}) \geq Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) - Q(\boldsymbol{\theta}, \boldsymbol{\theta})$$

where

$$Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) = \sum_{i=1}^{n} \sum_{m=1}^{M} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta}) \log \left(p(\mathbf{x}_i, \omega_m|\hat{\boldsymbol{\theta}}) \right)$$

which is, as stated in the question, the auxiliary function. [30%]

Significance. The difference $Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) - Q(\boldsymbol{\theta}, \boldsymbol{\theta})$ gives a lower bound on the increase in the log likelihood. Hence if we maximise the value of $Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$ the value of the log likelihood lower bound will also be maximised. This yields a global maximum of $Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$ as a closed form solution. This in turn leads to a natural iterative form of E-M where a new auxiliary function is constructed on each iteration. [10%]

(d) To get the new estimate we need to differentiate the auxiliary function wrt to the particular parameter of interest and equate to zero. In this case differentiate wrt $\boldsymbol{\mu}_m$. For Gaussian Mixture Models, the log likelihood for mixture component ω_m (d-dimensional data) is

$$\log\left(p(\mathbf{x}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)\right) = -\frac{1}{2}\left(\log((2\pi)^d|\boldsymbol{\Sigma}_m|) + (\mathbf{x} - \boldsymbol{\mu}_m)'\boldsymbol{\Sigma}_m^{-1}(\mathbf{x} - \boldsymbol{\mu}_m)\right)$$

The auxiliary function may be written as

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) = \quad & \sum_{m=1}^{M}\left[\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})\left(-\frac{1}{2}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_m)'\hat{\boldsymbol{\Sigma}}_m^{-1}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_m)\right)\right] \\ & + \sum_{m=1}^{M}\left[\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})\left(-\frac{1}{2}\log((2\pi)^d|\hat{\boldsymbol{\Sigma}}_m|)\right)\right] + \sum_{m=1}^{M}\left[\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})\log P(\omega_m)\right]\end{aligned}$$

where $\hat{\boldsymbol{\mu}}_m$ and $\hat{\boldsymbol{\Sigma}}_m$ are the re-estimated mean and covariance matrix of mixture component ω_m. Differentiating wrt $\boldsymbol{\mu}_m$ and equating to zero yields

$$\frac{\partial \mathcal{Q}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})}{\partial \boldsymbol{\mu}_m} = \sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})\hat{\boldsymbol{\Sigma}}_m^{-1}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_m) \quad = \quad 0$$

This givess the re-estimation formulae for the mean vector of component ω_m [20%]

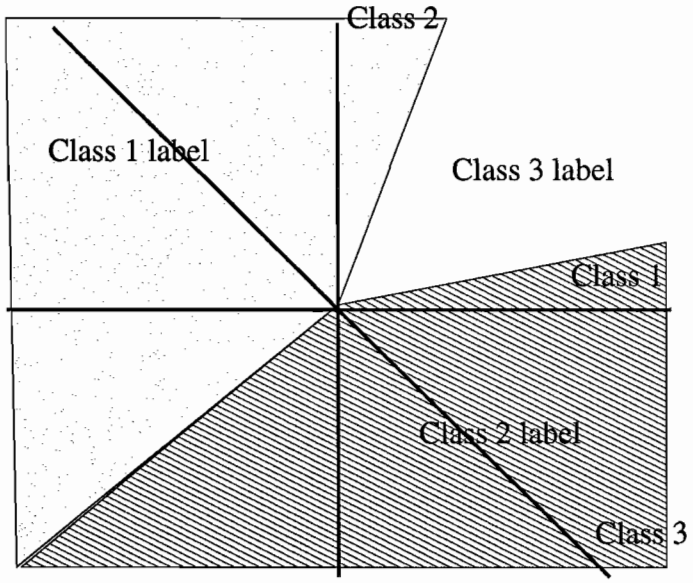
$$\hat{\boldsymbol{\mu}}_m = \frac{\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})\mathbf{x}_i}{\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\theta})}$$

(e) First need to decide on the number of Gaussian mixture components; initialise their values (to e.g. equal component priors, global covariance matrix and a set of different data points as values. Then iteratively update the mixture model parameters. Measure the log likelihood and check for convergence. A limit on the number of iterations may also be set.

Another issue (apart from number of components, initialisation and convergence) is that it is possible for the procedure to set the mean to a single training sample value with and the covariance will tend to zero and the likelihood to a maximum. To avoid this need either appropriate initialisation, to set a minimum on variance values, or don't use maximum likelihood estimation but incorporate a prior on Gaussian parameters. [20%]

3. *Kesler's construct and SVMs*

(a)(i) Dark line on diagram [20%]



(a)(ii) regions shown on diagram [30%]

(b) This is Kesler's construct. If $k = 1$ then

$$\mathbf{z}_2 = \begin{bmatrix} 1 \\ \mathbf{x} \\ -1 \\ -\mathbf{x} \\ 0 \\ \mathbf{0} \\ \vdots \end{bmatrix}, \quad \ldots \quad \mathbf{z}_K = \begin{bmatrix} 1 \\ \mathbf{x} \\ 0 \\ \mathbf{0} \\ \vdots \\ -1 \\ -\mathbf{x} \end{bmatrix}$$

Thus each observation is mapped in $K - 1$ points which all satisfy the inequality. [15%]

(c) (i) The following comments should be made

- The first issue to address is that there are no negative examples. These may be simply generated by producing extended vectors using examples from other classes. The form of the extended vector is the same (other than the shifting of the "class").
- The maximum margin criterion can be expressed as

$$\max_{\mathbf{w},b} \min \{||\mathbf{z} - \mathbf{z}_i||; \tilde{\mathbf{w}}'\mathbf{z} = 0, i = 1, \ldots, (K-1)Km\}$$

which may be expressed as (either form acceptable)

$$\min_{\mathbf{w},b} \tau(\tilde{\mathbf{w}}) = \frac{1}{2}||\tilde{\mathbf{w}}||^2$$

and all points satisfy

$$y_i \tilde{\mathbf{w}}' \mathbf{z}_i > 0$$

- SVMs are suitable for this training as the dimensionality of the data can become large as the extended vector is being used. As SVM are based on the maximum margin criterion which generalises well for high dimensional data. Also as the number of points scales the sparse representation of the SVM is efficient. [25%]

(c) (ii) When using an exponential kernel the effective dimensionality of the feature space is ∞. Thus an explicit representation in the feature space is not possible, so the decision boundary must be expressed in terms of the space spanned by the transformed points. So

$$\tilde{\mathbf{w}} = \sum_{i=1}^{(K-1)Km} \alpha_i \mathbf{z}_i$$

[10%]

4. *Decision Trees and ML criterion*

(a)(i) For the samples

$$\log\left(p(\mathbf{X}^{(p)}|\boldsymbol{\theta}^{(p)})\right) = \sum_{i=1}^{m} \log(\mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}^{(p)}, \boldsymbol{\Sigma}^{(p)})$$

[10%]

(a)(ii) As a diagonal covariance matrix is used deal with each dimension separately from (a)(i)

$$\log\left(p(\mathbf{X}^{(p)}|\boldsymbol{\theta}^{(p)})\right) = \sum_{i=1}^{m}\sum_{j=1}^{d}\left(-\log(\sqrt{2\pi\sigma_j^{(p)2}}) - \frac{(x_{ij}-\mu_j^{(p)})^2}{2\sigma_j^{(p)2}}\right)$$

Consider the quadratic term

$$\begin{aligned}
\sum_{i=1}^{m}\sum_{j=1}^{d}\frac{(x_{ij}-\mu_j^{(p)})^2}{2\sigma_j^{(p)2}} &= \sum_{i=1}^{m}\sum_{j=1}^{d}\frac{1}{2\sigma_j^{(p)2}}\left(x_{ij}^2 - 2x_{ij}\mu_j^{(p)} + \mu_j^{(p)2}\right) \\
&= m\sum_{j=1}^{d}\frac{1}{2\sigma_j^{(p)2}}\left(\sigma_j^{(p)2} + \mu_j^{(p)2} - 2\mu_j^{(p)2} + \mu_j^{(p)2}\right) \\
&= \frac{md}{2}
\end{aligned}$$

as the ML estimate of the parameters are used. Thus

$$\begin{aligned}\log\left(p(\mathbf{X}^{(p)}|\boldsymbol{\theta}^{(p)})\right) &= -m\log\left((2\pi)^{d/2}|\boldsymbol{\Sigma}^{(p)}|^{1/2}\right) - \frac{md}{2} \\ &= -\frac{md}{2} - \frac{md}{2}\log(2\pi) - \frac{m}{2}\log(|\boldsymbol{\Sigma}^{(p)}|)\end{aligned}$$

[30%]

(b) Change in log-likelihood from parent to the children

$$\Delta = -\frac{r}{2}\log(|\boldsymbol{\Sigma}^{(r)}|) - \frac{(m-r)}{2}\log(|\boldsymbol{\Sigma}^{(l)}|) + \frac{m}{2}\log(|\boldsymbol{\Sigma}^{(p)}|)$$

[20%]

(c) The stopping criterion is based on changes in log-likelihood. May also consider a minimum number of observations to try and generalise. Classification is most naturally based on looking at the most cpmmon class label associated with the node of the tree. This is the standard criterion. [20%]

(d) The Entropy measure is:

$$\mathcal{I}(N) = -\sum_{i=1}^{K} P(\omega_i|N)\log(P(\omega_i|N))$$

As it is directly based on classification performance (instead of the generative likelihood based approach) it should yiled gains. In terms of constructing the tree there is a choice of using the binary attributes to split or better performance may be obtained by using the standard continuous decision tree generation process - but this is expensive as all elements of all training vectors are examined. [20%]

5. *Gaussian Processes*

(a) From lecture notes the defintion is:

> A Gaussian Process is a collection of random variables, any finite number of which have Gaussian distributions. [10%]

(b)(i) The exponential kernel allows a non-linear function of the observations to be used for prediction. The value of σ^2 influences how smooth the prediction is. The larger the value of σ^2 the smoother the intyerpolation function (more points influence the prediction). [20%]

(b)(ii) By inspecition the mean is zero (the noise ϵ and w_i distributions are both zero. As in the lecture notes rewrite the interpolation as

$$f(\mathbf{x}) = \boldsymbol{\Phi}\mathbf{w}$$

wherethe $N \times H$ matrix, $\mathbf{\Phi}$ is

$$\mathbf{\Phi} = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \dots & \phi_H(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \phi_1(\mathbf{x}_n) & \dots & \phi_H(\mathbf{x}_n) \end{bmatrix}$$

Now we can write

$$\begin{aligned} \mathbf{y}\mathbf{y}' &= \mathbf{\Phi}\mathbf{w}\mathbf{w}'\mathbf{\Phi}' + \sigma_\epsilon^2\mathbf{I} \\ &= \sigma_{\mathrm{w}}^2\mathbf{\Phi}\mathbf{\Phi}' + \sigma_\epsilon^2\mathbf{I} \end{aligned}$$

So the i, j element of this matrix can be written as

$$[\mathbf{y}\mathbf{y}']_{ij} = \sigma_{\mathrm{w}}^2 \sum_{h=1}^{H} \exp\left(-\frac{(x_i - \mu_h)^2}{2\sigma^2}\right) \exp\left(-\frac{(x_j - \mu_h)^2}{2\sigma^2}\right) + \sigma_\epsilon^2\delta(i-j)$$

[25%]

(b)(iii) Now let $H \to \infty$ and $\sigma_{\mathrm{w}}^2 = \sigma_{\mathrm{h}}^2/H$. Thus need the limit of

$$[\mathbf{y}\mathbf{y}']_{ij} = \frac{\sigma_{\mathrm{h}}^2}{H} \sum_{h=1}^{H} \exp\left(-\frac{(x_i - \mu_h)^2}{2\sigma^2}\right) \exp\left(-\frac{(x_j - \mu_h)^2}{2\sigma^2}\right) + \sigma_\epsilon^2\delta(i-j)$$

This is in the form where

$$\begin{aligned} [\mathbf{y}\mathbf{y}']_{ij} &= \sigma_{\mathrm{h}}^2 \int \exp\left(-\frac{(x_i - \mu_h)^2}{2\sigma^2}\right) \exp\left(-\frac{(x_j - \mu_h)^2}{2\sigma^2}\right) p(\mu_h) d\mu_h + \sigma_\epsilon^2\delta(i-j) \\ &= \frac{\sigma_{\mathrm{h}}^2}{\sqrt{2\pi\sigma_{\mathrm{h}}^2}} \int \exp\left(-\frac{1}{2\sigma^2}\left((x_i - \mu_h)^2 + (x_j - \mu_h)^2\right) - \frac{1}{2\sigma_{\mathrm{h}}^2}\mu_h^2\right) d\mu_h + \sigma_\epsilon^2\delta(i-j) \\ &= \frac{\sigma_{\mathrm{h}}^2}{\sqrt{2\pi\sigma_{\mathrm{h}}^2}} \int \exp\left(-\frac{2\sigma_{\mathrm{h}}^2 + \sigma^2}{2\sigma^2\sigma_{\mathrm{h}}^2}\mu_h^2 + \frac{2(x_i + x_j)\mu_h}{2\sigma^2} - \frac{(x_i^2 + x_j^2)}{2\sigma^2}\right) d\mu_h + \sigma_\epsilon^2\delta(i-j) \\ &\approx \frac{\sigma_{\mathrm{h}}^2}{\sqrt{2\pi\sigma_{\mathrm{h}}^2}} \int \exp\left(-\frac{1}{\sigma^2}\mu_h^2 + \frac{2(x_i + x_j)\mu_h}{2\sigma^2} - \frac{(x_i^2 + x_j^2)}{2\sigma^2}\right) d\mu_h + \sigma_\epsilon^2\delta(i-j) \end{aligned}$$

Completing the square on this yields

$$[\mathbf{y}\mathbf{y}']_{ij} = \sqrt{\frac{\sigma^2\sigma_{\mathrm{h}}^2}{2}} \exp\left(-\frac{1}{4\sigma^2}(x_i - x_j)^2\right) + \sigma_\epsilon^2\delta(i-j)$$

[30%]

(b)(iv) The interpolation process can be expressed as a Gaussian process of the observations only. Thus though the number of weights will scale linearly with the number of basis, the power of the interpolation process only increases with the number of training examples, n. [15%]