Solutions to 4F10 Pattern Processing, 2009

1. *Bayes' Decision rule and generative models*

(a) Bayes' decision rule states

$$\text{Decide } \arg\max_{\omega_j} \{P(\omega_j|\boldsymbol{x})\}$$

which can be expressed for the generative classifiers here as

$$\text{Decide } \arg\max_{\omega_j} \{p(\boldsymbol{x}|\omega_j)P(\omega_j)\}$$

Generative models are minimum error classifiers is if there is

- infinite training data
- correct models (likelihood and priors)
- appropriate training algorithm

For most practical tasks none of these are true. [15%]

(b)(i) As the covariance matrices are diagonal each dimension can be treated separately. Thus need to maximise

$$\sum_{i=1}^{n} -\frac{y_i}{2}\left(\log(\sigma) + \frac{(x_i-\mu_1)^2}{\sigma^2}\right) - \frac{(1-y_i)}{2}\left(\log(\sigma) + \frac{(x_i-\mu_2)^2}{\sigma^2}\right)$$

Differentiating with respect to σ^2 yields

$$-\frac{n}{2\sigma^2} + \frac{1}{2}\sum_{i=1}^{n}\left(y_i\frac{(x_i-\mu_1)^2}{\sigma^4} + (1-y_i)\frac{(x_i-\mu_2)^2}{\sigma^4}\right)$$

Equating this to zero yields

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\left(y_i(x_i-\mu_1)^2 + (1-y_i)(x_i-\mu_2)^2\right)$$

[20%]

(b)(ii) The posterior for class ω_1 can be written as

$$\begin{aligned} P(\omega_1|\mathbf{x}) &= \frac{P(\omega_1)\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma})}{P(\omega_1)\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma}) + P(\omega_2)\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma})} \\ &= \frac{1}{1+\frac{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma})}{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma})}} \end{aligned}$$

It is possible to write

$$\begin{aligned}\frac{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_2,\boldsymbol{\Sigma})}{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma})} &= \exp\left((\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)'\boldsymbol{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}(\boldsymbol{\mu}_2'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2-\boldsymbol{\mu}_1'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1)\right)\\ &= \exp(-\mathbf{w}'\mathbf{x}+a)\end{aligned}$$

Thus

$$\begin{aligned}\mathbf{w} &= -(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)'\boldsymbol{\Sigma}^{-1}\\ a &= -\frac{1}{2}(\boldsymbol{\mu}_2'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2-\boldsymbol{\mu}_1'\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1)\\ \boldsymbol{\phi}(\mathbf{x}) &= \mathbf{x}\end{aligned}$$

[25%]

(c) Exactly the same form as before except the ratio will now be

$$\frac{\mathcal{N}(\mathbf{x};\mathbf{0},\boldsymbol{\Sigma}_2)}{\mathcal{N}(\mathbf{x};\mathbf{0},\boldsymbol{\Sigma}_1)} = \exp\left(-\frac{1}{2}\left(\mathbf{x}'\boldsymbol{\Sigma}_2^{-1}\mathbf{x}-\mathbf{x}'\boldsymbol{\Sigma}_1^{-1}\mathbf{x}+\log(|\boldsymbol{\Sigma}_2|)-\log(|\boldsymbol{\Sigma}_1|)\right)\right)$$

As the covariance matrices are both diagonal it is possible to write

$$\begin{aligned}\frac{1}{2}\left(\mathbf{x}'\boldsymbol{\Sigma}_2^{-1}\mathbf{x}-\mathbf{x}'\boldsymbol{\Sigma}_1^{-1}\mathbf{x}\right) &= \frac{1}{2}\sum_{i=1}^{d}x_i^2\left(\frac{1}{\sigma_{2i}^2}-\frac{1}{\sigma_{1i}^2}\right)\\ &= \mathbf{w}'\boldsymbol{\phi}(\mathbf{x})\end{aligned}$$

where

$$\begin{aligned}\mathbf{w} &= \frac{1}{2}\left[\ (\tfrac{1}{\sigma_{21}^2}-\tfrac{1}{\sigma_{11}^2})\ \ \ldots\ \ (\tfrac{1}{\sigma_{2d}^2}-\tfrac{1}{\sigma_{1d}^2})\ \right]'\\ a &= -\frac{1}{2}\log(|\boldsymbol{\Sigma}_2|)-\log(|\boldsymbol{\Sigma}_1|)\\ \boldsymbol{\phi}(\mathbf{x}) &= \left[\ x_1^2\ \ \ldots\ \ x_d^2\ \right]'\end{aligned}$$

[25%]

(d) The decision bioundary in (b) is a linear decision boundary. The form in part (c) is quadratic and results in an ellipsis centred around the origin. [15%]

2. *GMM Expectation Maximisation and Sequential Updates*

(a) Log-likelihood of the training data is

$$\log(p(x_1,\ldots,x_N|\theta)) = \sum_{i=1}^{N} \log\left(\sum_{m=1}^{M} c_m \mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m)\right)$$

[10%]

(b)(i) EM is an iterative approach to estimating the model parameters. Given the current estimates of the model parameters, $\boldsymbol{\theta}$, the new estimates, $\hat{\boldsymbol{\theta}}$, are found using

- Compute component posteriors, $P(\omega_m|x_i,\boldsymbol{\theta})$, using current parameters.
- Using the Auxiliary function, $\mathcal{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}})$, compute the new parameters.

[15%]

(b)(ii) Substituting in the expression for the likelihood to the auxiliary function

$$\mathcal{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}\sum_{m=1}^{M} P(\omega_m|x_i,\boldsymbol{\theta}) \log\left(\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m)\right)$$

Differentiate this with respect to $\boldsymbol{\mu}_q$ gives

$$\frac{\partial \mathcal{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\mu}}_q} = \sum_{i=1}^{n} P(\omega_q|x_i,\boldsymbol{\theta})\left[\hat{\boldsymbol{\Sigma}}^{-1}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_q)\right]$$

Equating to zero gives

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

[30%]

(c)(i) The sequential estimate can be expressed as

$$\begin{aligned}
\boldsymbol{\mu}_m^{(n)} &= \boldsymbol{\mu}_m^{(n-1)} + \left(\boldsymbol{\mu}_m^{(n)} - \boldsymbol{\mu}_m^{(n-1)}\right) \\
&= \boldsymbol{\mu}_m^{(n-1)} + \frac{b_{n-1}\mathbf{a}_n - b_n\mathbf{a}_{n-1}}{b_n b_{n-1}} \\
&= \boldsymbol{\mu}_m^{(n-1)} + \frac{b_{n-1}P(\omega_m|\mathbf{x}_n,\boldsymbol{\theta})\mathbf{x}_n - P(\omega_m|\mathbf{x}_n,\boldsymbol{\theta})\mathbf{a}_{n-1}}{b_n b_{n-1}} \\
&= \frac{P(\omega_m|\mathbf{x}_n,\boldsymbol{\theta})}{b_n}\left(\mathbf{x}_n - \boldsymbol{\mu}_m^{(n-1)}\right)
\end{aligned}$$

where

$$\mathbf{a}_n = \sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i,\boldsymbol{\theta})\mathbf{x}_i$$

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

thus

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

[30%]

(c)(i) The problem with the exact form is that the denominator for $\eta_m^{(n)}$ is a function of all the observations and the current model parameters. This means that every observation must be stored, making the update of little use for a sequential update. This is not the case for the approximate form. The motivation for the approximate form is that at the correct solution (using n observations)

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

[15%]

3. *Gaussian Processes and Automatic Relevance Detection*

(a) Basis linear regression, though allowing non-linear prediction, does not incorporate any concept of thge confidnece in the prediction. [10%]

(b) Interested in the joint distribution (directly from lecture notes)

$$\begin{bmatrix} f(\boldsymbol{x}) \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(\boldsymbol{x},\boldsymbol{x}) & \mathbf{k}(\boldsymbol{x},\boldsymbol{X})' \\ \mathbf{k}(\boldsymbol{x},\boldsymbol{X}) & \mathbf{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_\epsilon^2\mathbf{I} \end{bmatrix}\right)$$

Using the equality given in the question

$$p(f(\boldsymbol{x})|\boldsymbol{y},\boldsymbol{X}) = \mathcal{N}(f(\boldsymbol{x}); \mathbf{k}(\boldsymbol{x},\boldsymbol{X})'(\mathbf{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_\epsilon^2\mathbf{I})^{-1}\mathbf{y}; k(\boldsymbol{x},\boldsymbol{x}) - \mathbf{k}(\boldsymbol{x},\boldsymbol{X})'(\mathbf{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_\epsilon^2\mathbf{I})^{-1}\mathbf{k}(\boldsymbol{x},\boldsymbol{X}))$$

Again to get the distribution of y the noise variance is simply added. So

$$\begin{aligned} c &= k(\boldsymbol{x},\boldsymbol{x}) \\ \mathbf{d} &= \mathbf{k}(\boldsymbol{x},\boldsymbol{X}) \\ \mathbf{E} &= \mathbf{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_\epsilon^2\mathbf{I} \end{aligned}$$

[30%]

(c) Simplest approach to get this is to note that the marginal likelihood is Gaussian (as all aspects Gaussian). Only need the mean and variance.

$$\begin{aligned} \boldsymbol{\mu}_y &= \mathcal{E}\left\{\begin{bmatrix} f(\mathbf{x}_1) + \epsilon \\ \vdots \\ f(\mathbf{x}_n) + \epsilon \end{bmatrix}\right\} = \mathbf{0} \\ \boldsymbol{\Sigma}_y &= \mathcal{E}\left\{\boldsymbol{y}\boldsymbol{y}'\right\} \\ &= \mathcal{E}\left\{(\mathbf{f}(\mathbf{X})\boldsymbol{w} + \epsilon)(\mathbf{f}(\mathbf{X}) + \epsilon)'\right\} \\ &= \mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2\mathbf{I} \end{aligned}$$

This can be directly read from the joint distribution from part (b). [15%]

(d)(i) The hyperparameters can be estimated by maximising the marginal likelihood given in part (c) [10%]

(d)(ii) Looking at the value of $k(\mathbf{x}_i,\mathbf{x}_j)$ (assuming that dimension $m = d$ gives

$$\begin{aligned} k(\mathbf{x}_i,\mathbf{x}_j) &= \exp\left(-\frac{1}{2}\sum_{m=1}^{d}\frac{(x_{im} - x_{jm})^2}{\sigma_m^2}\right) \\ &= \exp\left(-\frac{1}{2}\sum_{m=1}^{d-1}\frac{(x_{im} - x_{jm})^2}{\sigma_m^2}\right) \end{aligned}$$

If $k(\mathbf{x}_i,\mathbf{x}_j)$ is now not a function of the d dimension, then neither will any of the elements from the prediction process in part (b). [20%]

(d)(iii) Using this form of covariance matrix the number of dimensions required to be computed for regression can be reduced (automatic relevance detection). In contrast for the form of RVM given in lectures only relevant training examples are selected for the regression process. [15%]

4. *Non-Parametric Techniques and Parzen Windows*

(a) The window function is valid PDF so

$$\int_{\mathcal{R}^d} \phi(\mathbf{x}) d\mathbf{x}; \quad \phi(\mathbf{x}) \geq 0$$

By simple analogy with the hypercube (or consider scaling each dimension)

$$\int_{\mathcal{R}^d} \phi(\frac{\mathbf{x}}{h}) d\mathbf{x} = h^d$$

Hence

$$\begin{aligned} \int_{\mathcal{R}^d} \tilde{p}(\mathbf{x}) d\mathbf{x} &= \int_{\mathcal{R}^d} \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h^d} \phi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) d\mathbf{x} \\ &= \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h^d} \int_{\mathcal{R}^d} \phi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) d\mathbf{x} \\ &= 1 \end{aligned}$$

[15%]

(b) Straight from the examples paper

$$\begin{aligned} \mathcal{E}\{\tilde{p}(x)\} &= \mathcal{E}\left\{\frac{1}{n} \sum_{i=1}^{n} \frac{1}{h} \phi\left(\frac{x - x_i}{h}\right)\right\} \\ &= \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h} \mathcal{E}\left\{\phi\left(\frac{x - x_i}{h}\right)\right\} \\ &= \frac{1}{h} \mathcal{E}\left\{\phi\left(\frac{x - x_i}{h}\right)\right\} \end{aligned}$$

As $\phi()$ is a Gaussian function then

$$\begin{aligned} \mathcal{E}\left\{\mathcal{N}\left(\frac{x - x_i}{h}; 0, 1\right)\right\} &= \int \mathcal{N}\left(\left(\frac{x - v}{h}\right); 0, 1\right) \mathcal{N}(v; \mu, \sigma^2) dv \\ &= \int h\mathcal{N}(x; v, h^2) \mathcal{N}(v; \mu, \sigma^2) dv \\ &= h\mathcal{N}(x; \mu, \sigma^2 + h^2) \end{aligned}$$

Hence

$$\mathcal{E}\{\tilde{p}(x)\} = \mathcal{N}(x; \mu, \sigma^2 + h^2)$$

[30%]

(c)(i) The form of Gaussian window function and the first order Taylor series expansion is

$$\begin{aligned} \phi\left(\frac{x - x_i}{h}\right) &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x - x_i)^2}{2h^2}\right) \\ &\approx \frac{1}{\sqrt{2\pi}} \left(1 - \frac{(x - x_i)^2}{2h^2}\right) \end{aligned}$$

The approximate Parzen window is then

$$\begin{aligned}\tilde{p}(x) &\approx \frac{1}{hn}\sum_{i=1}^{n}\left(\frac{1}{\sqrt{2\pi}}\left(1-\frac{(x-x_i)^2}{2h^2}\right)\right)\\ &= \frac{1}{hn\sqrt{2\pi}}\sum_{i=1}^{n}\left(1-\frac{x^2}{2h^2}+\frac{xx_i}{h^2}-\frac{x_i^2}{2h^2}\right)\end{aligned}$$

[25%]

(c)(ii) It is only necessary to store the values of b_0, b_1 and b_2. The Parzen window approximation is then simply computed as calculating the weighted sum for the Taylor series. The approximation requires that

$$\frac{x-x_i}{h} << 1$$

This requires that value of h should be large. However this results in a poor approximation for the true distribution $p(x)$. [15%]

(c)(iii) Taking the expected values

$$\begin{aligned}\mathcal{E}\left\{\frac{1}{hn\sqrt{2\pi}}\sum_{i=1}^{n}\left(1-\frac{x^2}{2h^2}+\frac{xx_i}{h^2}-\frac{x_i^2}{2h^2}\right)\right\} &= \frac{1}{h\sqrt{2\pi}}\left(1-\frac{x^2}{2h^2}+\frac{x\mathcal{E}\{x_i\}}{h^2}-\frac{\mathcal{E}\{x_i^2\}}{2h^2}\right)\\ &= \frac{1}{h\sqrt{2\pi}}\left(1-\frac{x^2}{2h^2}+\frac{x\mu}{h^2}-\left(\frac{\sigma^2+\mu^2}{2h^2}\right)\right)\\ &= \frac{1}{h\sqrt{2\pi}}\left(1-\frac{(x-\mu)^2}{2h^2}-\frac{\sigma^2}{2h^2}\right)\end{aligned}$$

As h becomes increasingly large as in part (c)(ii) this will be an approximation to the expected value of part (b). [15%]

5. *Support Vector Machines and Kernels*

(a) The training criterion for the perceptron algorithm is the sum of the perpendicular distance from the decision boundary of all miclassified points. There is no unique solution to this cost function. For the SVM classifier the cost function is to maximum the margin, i.e.

$$\max_{\mathbf{w},b} \min \{||\mathbf{x} - \mathbf{x}_i||; \langle \mathbf{w}, \mathbf{x} \rangle + b = 0, i = 1, \ldots, m\}$$

This has the dual form which required minimising

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

subject to

$$y_i \left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b \right) \geq 1$$

for all $i = 1, \ldots, m$.

When the data is not linearly separable *slack variables* must be introduced. The soft-margin classifier is obtained by minimising

$$\tau(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{m} \xi_i$$

subject to

$$\begin{aligned} y_i \left(\langle \mathbf{w}, \mathbf{x}_i \rangle + b \right) &\geq 1 - \xi_i \\ \xi_i &\geq 0 \end{aligned}$$

[25%]

(b) Kernels map from the input-space to a higher dimensional feature-space. A linear classifier is then built in this feature space. This results in a non-linear decision boundary in the original feature space. The general form for the polynomial kernel is

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i . \mathbf{x}_j + 1)^d$$

[15%]

(c)(i) For the in-homogeneous polynomial kernel the feature-space has the form (others are possible, but this is the most informative)

$$\boldsymbol{\Phi}(x) = \begin{bmatrix} a_0 & a_1 x & a_2 x^2 & \ldots & a_N x^N \end{bmatrix}'$$

where the values of $a_0, \ldots, a_N$ are determined by the order of the polynomial. There are thus two main differences

(a) the kernel operates on the exponential of the input-space

(b) there are no scaling factors - depending on the metric used this may or not affect the decision boundary generated. [20%]

(c)(ii) The kernel function is found from noting that the dot product is a geometric progression

$$\begin{aligned} k(x_i, x_j) &= 1 + \sum_{r=1}^{N} \exp(r(x_i + x_j)) \\ &= \frac{1 - e^{(N+1)(x_i+x_j)}}{1 - e^{(x_i+x_j)}} \end{aligned}$$

[25%]

(d) The kernelised version of the rule is

$$g_i(x) = \sum_{i=1}^{m} y_i \alpha_i k(x, x_i) + b$$

Once the number of support vectors are known then the computational cost is independent of the number of training samples and N. The cost scales linearly with the number of support vectors. [15%]