PART IIB 2010 DR MJF GALES

Solutions to 4F10 Pattern Processing, 2010

1. *Bayes' Decision rule and generative models*

(a) This is direct from the lecture notes. The log-likelihood for class ω_1 can be written as

$$\log(p(\boldsymbol{x}|\omega_1)) = -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}^{(1)})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}^{(1)}) + \log\left(\frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}}\right)$$

The final term is a constant (i.e. independent of $\boldsymbol{x}$) A point lying on the decision boundary satisfies

$$\left(\boldsymbol{\mu}^{(1)}-\boldsymbol{\mu}^{(2)}\right)'\boldsymbol{\Sigma}^{-1}\boldsymbol{x} = \frac{1}{2}\left(\boldsymbol{\mu}^{(1)\prime}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}^{(1)} - \boldsymbol{\mu}^{(2)\prime}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}^{(2)}\right)$$

The equation of this class boundary is of the form: $\mathbf{w}'\,\boldsymbol{x} = b$. The inverse of the covariance matrix is given by

$$\boldsymbol{\Sigma}^{-1} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}$$

Plugging the values in yields

$$\begin{bmatrix} -2 \\ 2 \end{bmatrix}'\boldsymbol{x} = 2; \quad x_2 - x_1 = 1$$

[35%]

(b) When the covariance matrix is restricted to be the identity matrix the solution will simply be half-way between the two means

$$x_1 = 0$$

[15%]

(c) (i) The means of the two classes can be derived directly from the Gaussians above. The mean in the transformed space will be

$$\mathcal{E}\{\boldsymbol{y}\} = \begin{bmatrix} 1 & \mu_1 & \mu_2 & \sigma_1^2+\mu_1^2 & \sigma_{12}+\mu_1\mu_2 & \sigma_2^2+\mu_2^2 \end{bmatrix}'$$

Substituting the values into this expression yields

$$\begin{aligned} \tilde{\boldsymbol{\mu}}^{(1)} &= \begin{bmatrix} 1 & -1 & 1 & 3 & 0 & 2 \end{bmatrix}' \\ \tilde{\boldsymbol{\mu}}^{(2)} &= \begin{bmatrix} 1 & 1 & 1 & 3 & 2 & 2 \end{bmatrix}' \end{aligned}$$

The decision boundary, using identity matrices will be given by

$$\begin{bmatrix} 0 & -2 & 0 & 0 & -2 & 0 \end{bmatrix}'\boldsymbol{y} = -2$$

This can be expressed in the original-space as

$$x_1 + x_1 x_2 = 1$$

[30%]

(c) (ii) The optimal decision boundary for this problem was derived in (a). This can be rewritten as

$$x_2 - x_1 - 1 = 0$$

Thus the optimal decision boundary will be (a possibly scaled version of)

$$\begin{bmatrix} -1 & -1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

[10%]

(c) (iii) If the Gaussians have a general form of covariance matrix (no constraints on them being equal). This will yield a quadratic form of optimal decision boundary of the form

$$\boldsymbol{x}'\mathbf{A}\boldsymbol{x} + \mathbf{b}'\boldsymbol{x} + c = 0$$

This can be exactly modelled with a linear decision boundary in the $\boldsymbol{y}$ space. [10%]

Assessor's comment:

This question looked at Bayes' decision rule and non-linear transformations. Most candidates understood Bayes' decision rule and the decision boundaries resulting from class-conditional Gaussian distributions. However, many candidates did not realise that, for non-linear transforms, the mean in the transformed space is not simply the transformed mean.

2. *M*ixture Models and the Exponential Family

(a) Z must satisfy

$$Z = \int \exp\left(\boldsymbol{\alpha}' \mathbf{f}(\mathbf{x})\right) dx$$

this ensures that the PDF integrates to 1. [10%]

(b) It is possible to rewrite the expression as

$$\begin{aligned} p(\mathbf{x}|\boldsymbol{\mu}) &= \frac{1}{Z} \prod_{i=1}^{d} \mu_i^{x_i} \\ &= \frac{1}{Z} \exp\left(\sum_{i=1}^{d} x_i \log(\mu_i) \right) \end{aligned}$$

By analogy with the exponential distribution

$$\lambda_i = -\log(\mu_i)$$

The values are then

$$\begin{aligned} \boldsymbol{\alpha} &= \begin{bmatrix} \log(\mu_1) \\ \vdots \\ \log(\mu_d) \end{bmatrix} \\ \mathbf{f}(\mathbf{x}) &= \mathbf{x} \\ Z &= \frac{1}{\prod_{i=1}^{d} \left(-\log(\mu_d)\right)} \end{aligned}$$

[30%]

(c)(i) The expression for the log-likelihood is

$$l(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \log \left(\sum_{m=1}^{M} c_m \frac{1}{Z_m} \exp\left(\boldsymbol{\alpha}_m' \mathbf{f}(x_i)\right) \right)$$

[15%]

(c)(ii) Substituting in the expression for the exponential family

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}) &= \sum_{i=1}^{n} \sum_{m=1}^{M} P(\omega_m|\mathbf{x}_i, \boldsymbol{\alpha}) \left[-\log(\hat{Z}_m) + \hat{\boldsymbol{\alpha}}_m \mathbf{f}(\mathbf{x}_i) \right] \\ &= \sum_{m=1}^{M} \left(-\log(\hat{Z}_m) \left(\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\alpha}) \right) + \hat{\boldsymbol{\alpha}}_m \left(\sum_{i=1}^{n} P(\omega_m|\mathbf{x}_i, \boldsymbol{\alpha}) \mathbf{f}(\mathbf{x}) \right) \right) \end{aligned}$$

Sufficient statistics for auxiliary function is simply

$$\sum_{i=1}^{n} P(\omega_m|x_i, \boldsymbol{\alpha}); \quad \sum_{i=1}^{n} P(\omega_m|x_i, \boldsymbol{\alpha}) \mathbf{f}(\mathbf{x}_i)$$

for each of the components of the model. [25%]

(c)(ii) Differentiating the auxiliary function yields

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\alpha}_m}\mathcal{Q}(\boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}) &= \sum_{i=1}^{N} P(\omega_m|\mathbf{x}_i, \boldsymbol{\alpha}) \frac{\partial}{\partial \boldsymbol{\alpha}_m} \left(\log(c_m) - \log(Z_m) + \boldsymbol{\alpha}_m' \mathbf{f}(\mathbf{x}_i)\right) \\ &= \sum_{i=1}^{N} P(\omega_m|\mathbf{x}_i, \boldsymbol{\alpha}) \left[-\frac{1}{Z_m}\frac{\partial}{\partial \boldsymbol{\alpha}_m} Z_m + \mathbf{f}(\mathbf{x}_i)\right]\end{aligned}$$

In the general case the normalisation term will not be linear, so general optimisation approaches are required, for example gradient descent. [20%]

Assessor's comments:

This questioned assessed candidates knowledge of mixtures models. The first section was well answered. However many candidates did not find all the statistics that needed to be extracted from the training data to estimate the model parameters. Overall candidates performed well on this question.

3. *Gaussian Processes Regression*

(a) Basis function regression, though allowing non-linear prediction, does not incorporate any concept of the confidence 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)(i) [This is based on an examples paper question]. From (b)

$$\begin{aligned} \mu_{\mathrm{f}} &= \mathbf{k}(\tilde{\mathbf{x}},\mathbf{X})'\left[\mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_{\mathrm{n}}^2\mathbf{I}\right]^{-1}\mathbf{y} \\ \mathrm{var}_n(f(\tilde{\mathbf{x}})) &= \sigma_{\mathrm{f}}^2 = k(\tilde{\mathbf{x}},\tilde{\mathbf{x}}) - \mathbf{k}(\tilde{\mathbf{x}},\mathbf{X})'\left[\mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_{\mathrm{n}}^2\mathbf{I}\right]^{-1}\mathbf{k}(\tilde{\mathbf{x}},\mathbf{X}) \end{aligned}$$

Now consider the following partition where $\bar{\mathbf{X}}$ excludes observation $\mathbf{x}_n$ from the training data

$$\left[\mathbf{K}(\mathbf{X},\mathbf{X}) + \sigma_{\mathrm{n}}^2\mathbf{I}\right]^{-1} = \begin{bmatrix} \mathbf{K}(\bar{\mathbf{X}},\bar{\mathbf{X}}) + \sigma_{\mathrm{n}}^2\mathbf{I} & \mathbf{k}(\mathbf{x}_n,\bar{\mathbf{X}}) \\ \mathbf{k}(\mathbf{x}_n,\bar{\mathbf{X}})' & k(\mathbf{x}_n,\mathbf{x}_n) + \sigma_{\mathrm{n}}^2 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{b}' & c \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{E} & \mathbf{g} \\ \mathbf{g}' & h \end{bmatrix}$$

and

$$\mathbf{k}(\tilde{\mathbf{x}},\mathbf{X}) = \begin{bmatrix} \mathbf{k}(\tilde{\mathbf{x}},\bar{\mathbf{X}}) \\ \mathbf{k}(\tilde{\mathbf{x}},\mathbf{x}_n) \end{bmatrix}$$

Looking at the second term in the variance only, for the n samples this can be expressed as

$$d_n = \mathbf{k}(\tilde{\mathbf{x}},\bar{\mathbf{X}})'\mathbf{E}\mathbf{k}(\tilde{\mathbf{x}},\bar{\mathbf{X}}) + 2k(\tilde{\mathbf{x}},\mathbf{x}_n)\mathbf{g}'\mathbf{k}(\tilde{\mathbf{x}},\bar{\mathbf{X}}) + k(\tilde{\mathbf{x}},\mathbf{x}_n)hk(\tilde{\mathbf{x}},\mathbf{x}_n)$$

and for the $n-1$ samples

$$d_{n-1} = \mathbf{k}(\tilde{\mathbf{x}},\bar{\mathbf{X}})'(\mathbf{K}(\bar{\mathbf{X}},\bar{\mathbf{X}}) + \sigma_{\mathrm{n}}^2\mathbf{I})^{-1}\mathbf{k}(\tilde{\mathbf{x}},\bar{\mathbf{X}})$$

Consider the form for the partitioned n example case. Let

$$k_a = \mathbf{k}(\tilde{\mathbf{x}}, \bar{\mathbf{X}})' \mathbf{A}^{-1} \mathbf{b}$$

and

$$k_b = \frac{1}{k(\mathbf{x}_n, \mathbf{x}_n) + \sigma_{\mathrm{n}}^2 - \mathbf{k}(\mathbf{x}_n, \bar{\mathbf{X}})' \left(\mathbf{K}(\bar{\mathbf{X}}, \bar{\mathbf{X}}) + \sigma_{\mathrm{n}}^2 \mathbf{I}\right)^{-1} \mathbf{k}(\mathbf{x}_n, \bar{\mathbf{X}})}$$

Then

$$\begin{aligned} d_n &= d_{n-1} + k_a k_b k_a - 2k(\tilde{\mathbf{x}}, \mathbf{x}_n) k_b k_a + k(\tilde{\mathbf{x}}, \mathbf{x}_n) k_b k(\tilde{\mathbf{x}}, \mathbf{x}_n) \\ &= d_{n-1} + (k(\tilde{\mathbf{x}}, \mathbf{x}_n) - k_a)^2 k_b \end{aligned}$$

by definition the second term is non-negative (definition of semi-positive definite), so

$$d_n \geq d_{n-1} \quad \text{therefore} \quad \mathrm{var}_n(f(\tilde{\mathbf{x}})) \leq \mathrm{var}_{n-1}(f(\tilde{\mathbf{x}}))$$

[50%]

(c)(ii) The implications of this is that as the number of points for Gaussian process regression increases, the confidence in the prediction increases. [10%]

Assessor's comment:

An unpopular question The question looked at some standard properties of Gaussian processes. The question was reasonably well done by the candidates. However it was disappointing that the proof of the variances of the prediction was not better answered as this was very similar to an examples paper question.

4. *Support Vector Machines and Multi-Class Classification*

(a)(i) One of the two classes must be assigned for class 1 $\tilde{y}_i = 1$ and for class 2 $\tilde{y}_i = -1$. All points must lie beyond the margin. In this case all samples must satisfy

$$\tilde{y}_i(\mathbf{w}'\boldsymbol{\phi}(\mathbf{x}) + b) \geq 1$$

[15%]

(a)(ii) Points to be mentioned:

(a) The values of the Lagrange multipliers are found by maximising the margin.

(b) For each pair one is selected as being class 1, the second as class 2. The SVM for this pair is then trained. This allows $\alpha_i^{(pq)}$ to be found where

$$\alpha_i^{(pq)} = \tilde{y}_i \alpha_i$$

(c) For all training examples that don't belong to ω_p and ω_q $\alpha_i^{(pq)} = 0$. Effectively this will not be support vectors.

(d) The values of $b^{(pq)}$ are found in the standard fashion from the SVM. Training examples with non-zero support vectors are used and then points that satisfy

$$\tilde{y}_i(\mathbf{w}'\boldsymbol{\phi}(\mathbf{x}) + b) = 1$$

[20%]

(a)(iii) Two approaches mentioned in lectures, majority voting and use of adaptive DAG classification. The simplest is majority voting, discussed here.

- Each of the $K(K-1)/2$ classifiers are run.
- The class outcome from each of these is counted as a vote for each of them. There are therefore $K(K-1)/2$ SVM classification runs.
- A problem arises from "ties" where the votes for two or more of the classes are the same. For a tie between two classes is normally resolved using the classification result for these classes. For more than two classes it is not possible to guarantee that a clear winner will result.

[20%]

(b) (i) This is a standard form discussed in lectures, Kesler's constrict, but applied to SVMs. If observation $\mathbf{x}_i$ belongs to class ω_1 for example and $j = 2$

$$\mathbf{z}_j = \begin{bmatrix} 1 \\ \boldsymbol{\phi}(\mathbf{x}_i) \\ -1 \\ -\boldsymbol{\phi}(\mathbf{x}_i) \\ \mathbf{0} \end{bmatrix}$$

All $K-1$ possible values of j must be considered. [15%]

(b)(ii) To train the SVM both positive and negative examples are required. These can be simply generated by using setting one of the training examples as if it belonged to a different class to it's actual one. It is then possible to form the Gram matrix, which will be $(K-1)m \times (K-1)m$. [15%]

(b)(iii) The computational cost of this approach is to perform the kernel operations in the extended space defined by $\tilde{\mathbf{w}}$. The issues to consider are

- single classifier but the observation space is K times as large. However there are $K(K-1)/2$ classifications for a(iii).
- the number of support vectors. This is expected to be larger for the single classifier than each of the individual ones. [15%]

Assessor's comment:

The least popular question. The question examined the use of SVMs for multi-class classification. Though the candidates showed a good knowledge of SVMs and SVM training, it was disappointing that the extension to the multi-class case was not well explained.

5. *Training Logistic Regression and Regularisation*

(a) The log-probability of the data from class ω_1 can be written as

$$\begin{aligned}\mathcal{L}(\mathbf{b}) &= \sum_{i=1}^{n} (y_i \log(P(\omega_1|\mathbf{x}_i, \mathbf{b})) + (1 - y_i) \log(P(\omega_2|\mathbf{x}_i, \mathbf{b})) \\ &= \sum_{i=1}^{n} (y_i \log(P(\omega_1|\mathbf{x}_i, \mathbf{b})) + (1 - y_i) \log(1 - P(\omega_1|\mathbf{x}_i, \mathbf{b}))\end{aligned}$$

This will yield a linear decision boundaries passing through the origin. [15%]

(a)(ii) Differentiating

$$\begin{aligned}\frac{\partial}{\partial \mathbf{b}} P(\omega_1|\mathbf{x}, \mathbf{b}) &= \frac{\exp(-\mathbf{b}'\mathbf{x})}{(1 + \exp(-\mathbf{b}'\mathbf{x}))^2}\mathbf{x} \\ &= P(\omega_1|\mathbf{b}, \mathbf{x})(1 - P(\omega_1|\mathbf{b}, \mathbf{x}))\mathbf{x}\end{aligned}$$

Thus

$$\begin{aligned}\frac{\partial}{\partial \mathbf{b}} \mathcal{L}(\mathbf{b}) &= \sum_{i=1}^{n} \mathbf{x}_i \left(y_i(1 - P(\omega_1|\mathbf{b}, \mathbf{x}_i)) - (1 - y_i)P(\omega_1|\mathbf{b}, \mathbf{x}_i)\right) \\ &= \sum_{i=1}^{n} \mathbf{x}_i \left(y_i - P(\omega_1|\mathbf{b}, \mathbf{x}_i)\right)\end{aligned}$$

This can be used in a gradient style approach where

$$\mathbf{b}^{(k+1)} = \mathbf{b}^{(k)} + \eta \left.\frac{\partial}{\partial \mathbf{b}} \mathcal{L}(\mathbf{b})\right|_{\mathbf{b}^{(k)}}$$

[30%]

(a)(iii) Element j, k of the Hessian is

$$h_{jk} = \frac{\partial^2}{\partial b_j \partial b_k} \mathcal{L}(\mathbf{b})$$

The Hessian may be used for optimisation as

$$\mathbf{b}^{(k+1)} = \mathbf{b}^{(k)} + \mathbf{H}^{-1} \left.\frac{\partial}{\partial \mathbf{b}} \mathcal{L}(\mathbf{b})\right|_{\mathbf{b}^{(k)}}$$

where the Hessian is evaluated at the $\mathbf{b}^{(k)}$. [10%]

(b)(i) The form of regularisation will penalise terms that move away from having a value of zero. This prevents large values of $\mathbf{b}$ being trained. The value of λ determines the degree to which this term is used. [15%]

(b)(ii) This just requires an additional term to be added to the previous one

$$\frac{\partial}{\partial \mathbf{b}} \mathcal{F}(\mathbf{b}) = \frac{\partial}{\partial \mathbf{b}} \mathcal{L}(\mathbf{b}) - 2\lambda \mathbf{b}$$

[15%]

(b)(iii) The Hessian is simply altered by adding a term which is $-2\lambda\mathbf{I}$ where $\mathbf{I}$ is the identity matrix. One problem that can occur for maximising the likelihood is that Hessian is not necessarily negative semi-definite, or invertible (especially as d becomes large). By introducing the regularisation term with a large enough value of λ it is possible to ensure that both of these requirements are satisfied. [15%]

Assessor's comment:

This question examined the candidates knowledge of gradient descent based optimisation and the use of second-order approaches. The question was popular and generally well done with most candidates understanding the use of the Hessian and regularisation term.