

Stat 648: Assignment 2 Solutions (115 points)

(3.5) (5 pts.) Writing the ridge expression as

$$\begin{aligned} & \sum_{i=1}^N \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij}\beta_j - \sum_{j=1}^p \bar{x}_j\beta_j + \sum_{j=1}^p \bar{x}_j\beta_j \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 \\ &= \sum_{i=1}^N \left(y_i - (\beta_0 + \sum_{j=1}^p \bar{x}_j\beta_j) - \sum_{j=1}^p (x_{ij} - \bar{x}_j)\beta_j \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 \\ &= \sum_{i=1}^N \left(y_i - \beta_0^c - \sum_{j=1}^p (x_{ij} - \bar{x}_j)\beta_j^c \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 \quad \text{for } \beta_j^c = \beta_j \text{ and } \beta_0^c = \beta_0 + \sum_{j=1}^p \bar{x}_j\beta_j. \end{aligned}$$

and shifting the x_i 's to have zero mean only modifies the intercept and not the slopes.

A similar argument holds for the lasso by again letting $\beta_j^c = \beta_j$ and $\beta_0^c = \beta_0 + \sum_{j=1}^p \bar{x}_j\beta_j$.

(3.12) (5 pts.) With $\mathbf{X}^{aug} = \begin{bmatrix} \mathbf{X} \\ \sqrt{\lambda} \mathbf{I} \end{bmatrix}$ and $\mathbf{y}^{aug} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix}$,

$$\mathbf{y}'^{aug}\mathbf{y}^{aug} = [\mathbf{y}' \ \mathbf{0}'] \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix} = \mathbf{y}'\mathbf{y},$$

$$\mathbf{X}'^{aug}\mathbf{X}^{aug} = [\mathbf{X}' \ \sqrt{\lambda}\mathbf{I}] \begin{bmatrix} \mathbf{X} \\ \sqrt{\lambda}\mathbf{I} \end{bmatrix} = \mathbf{X}'\mathbf{X} + \lambda\mathbf{I},$$

and

$$\mathbf{y}'^{aug}\mathbf{X}^{aug} = [\mathbf{y}' \ \mathbf{0}'] \begin{bmatrix} \mathbf{X} \\ \sqrt{\lambda}\mathbf{I} \end{bmatrix} = \mathbf{y}'\mathbf{X}.$$

$$\text{So, } \hat{\boldsymbol{\beta}}^{OLS} = (\mathbf{X}'^{aug}\mathbf{X}^{aug})^{-1}\mathbf{X}'^{aug}\mathbf{y}^{aug} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{y} = \hat{\boldsymbol{\beta}}^{ridge}.$$

(3.16) (15 pts.) First, note that $\hat{\boldsymbol{\beta}}^{OLS} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} = \mathbf{X}'\mathbf{y}$ because of the orthonormality of \mathbf{X} .

Best subset: Since the inputs are orthogonal, dropping terms will not change the estimates of the other terms. Letting \mathbf{y}_M , \mathbf{X}_M , and $\boldsymbol{\beta}_M$ represent reduced version of \mathbf{y} , \mathbf{X} , and $\boldsymbol{\beta}$ of size M , we wish to minimize

$$\begin{aligned} (\mathbf{y}_M - \mathbf{X}_M\boldsymbol{\beta}_M)'(\mathbf{y}_M - \mathbf{X}_M\boldsymbol{\beta}_M) &= \mathbf{y}'_M\mathbf{y}_M - 2\mathbf{y}'_M\mathbf{X}_M\boldsymbol{\beta}_M + \boldsymbol{\beta}'_M\mathbf{X}'_M\mathbf{X}_M\boldsymbol{\beta}_M \\ &= \mathbf{y}'_M\mathbf{y}_M - \boldsymbol{\beta}'_M\boldsymbol{\beta}_M \text{ since } \mathbf{X}_M \text{ is orthonormal.} \end{aligned}$$

This is minimized by choosing the M largest β_j 's in magnitude, giving estimates $\hat{\beta}_j I[\text{rank}|\hat{\beta}_j| \leq M]$.

Ridge: By Eqn. (3.44),

$$\begin{aligned} \hat{\boldsymbol{\beta}}^{ridge} &= (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{y} \\ &= (\mathbf{I} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{y} \quad \text{since } \mathbf{X} \text{ is orthogonal} \\ &= ((1 + \lambda)\mathbf{I})^{-1}\mathbf{X}'\mathbf{y} \\ &= \frac{1}{1 + \lambda}\mathbf{X}'\mathbf{y} = \frac{1}{1 + \lambda}\hat{\boldsymbol{\beta}}^{OLS} \end{aligned}$$

Lasso: $\hat{\boldsymbol{\beta}}^{lasso} = \arg \min_{\boldsymbol{\beta}} \left\{ (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \lambda^* \sum_{j=1}^p |\beta_j| \right\}$ where $\lambda^* = 2\lambda$ and

$$\begin{aligned} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) &= \mathbf{y}'\mathbf{y} - 2\mathbf{y}'\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta} \\ &= \mathbf{y}'\mathbf{y} - 2\mathbf{y}'\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}'\boldsymbol{\beta} \\ &= \mathbf{y}'\mathbf{y} - 2\hat{\boldsymbol{\beta}}^{OLS}\boldsymbol{\beta} + \boldsymbol{\beta}'\boldsymbol{\beta} \\ &= \mathbf{y}'\mathbf{y} - 2\sum_{j=1}^p \hat{\beta}_j^{OLS}\beta_j + \sum_{j=1}^p \beta_j^2 \end{aligned}$$

Thus, we wish to minimize $f(\boldsymbol{\beta}) \equiv \mathbf{y}'\mathbf{y} - 2\sum_{j=1}^p \hat{\beta}_j^{OLS}\beta_j + \sum_{j=1}^p \beta_j^2 + 2\lambda \sum_{j=1}^p |\beta_j|$.

Setting $\frac{df(\boldsymbol{\beta})}{d\beta_j} = -2\hat{\beta}_j^{OLS} + 2\beta_j + 2\lambda \operatorname{sgn}(\beta_j) = 0$, and since $\hat{\beta}_j^{OLS}$ and β_j have the same signs,

$$\begin{aligned} \beta_j &= \hat{\beta}_j^{OLS} - \lambda \operatorname{sgn}(\hat{\beta}_j^{OLS}) \\ &= \operatorname{sgn}(\hat{\beta}_j^{OLS})(|\hat{\beta}_j^{OLS}| - \lambda)_+ \end{aligned}$$

(3.30) (8 pts.) Let $\mathbf{X}^{aug} = \begin{bmatrix} \mathbf{X} \\ \sqrt{\alpha\lambda} \mathbf{I} \end{bmatrix}$ and $\mathbf{y}^{aug} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix}$. Doing the lasso with \mathbf{X}^{aug} and \mathbf{y}^{aug} , we wish to find

$$\begin{aligned} &\min_{\boldsymbol{\beta}} \left\{ (\mathbf{y}^{aug} - \mathbf{X}^{aug}\boldsymbol{\beta})'(\mathbf{y}^{aug} - \mathbf{X}^{aug}\boldsymbol{\beta}) + \lambda^* \sum_{j=1}^p |\beta_j| \right\} \text{ where } \lambda^* = \lambda(1 - \alpha) \\ &= \min_{\boldsymbol{\beta}} \left\{ \mathbf{y}'^{aug}\mathbf{y}^{aug} - 2\mathbf{y}'^{aug}\mathbf{X}^{aug}\boldsymbol{\beta} + \boldsymbol{\beta}'\mathbf{X}'^{aug}\mathbf{X}^{aug}\boldsymbol{\beta} + \lambda^* \sum_{j=1}^p |\beta_j| \right\}. \end{aligned}$$

Now, in a similar manner to 3.12, $\mathbf{y}'^{aug}\mathbf{y}^{aug} = \mathbf{y}'\mathbf{y}$,

$$\mathbf{X}'^{aug}\mathbf{X}^{aug} = \mathbf{X}'\mathbf{X} + \alpha\lambda\mathbf{I},$$

$$\mathbf{y}'^{aug}\mathbf{X}^{aug} = \mathbf{y}'\mathbf{X}.$$

So, we have

$$\begin{aligned} &\min_{\boldsymbol{\beta}} \left\{ \mathbf{y}'\mathbf{y} - 2\mathbf{y}'\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}'(\mathbf{X}'\mathbf{X} + \alpha\lambda\mathbf{I})\boldsymbol{\beta} + \lambda(1 - \alpha) \sum_{j=1}^p |\beta_j| \right\} \\ &= \min_{\boldsymbol{\beta}} \left\{ (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \alpha\lambda\boldsymbol{\beta}'\boldsymbol{\beta} + \lambda(1 - \alpha) \sum_{j=1}^p |\beta_j| \right\} \\ &= \min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \left[\alpha \|\boldsymbol{\beta}\|_2^2 + (1 - \alpha) \|\boldsymbol{\beta}\|_1 \right]. \end{aligned}$$

(4.6) (10 pts.) (a) Separability implies that $\exists \boldsymbol{\beta}_{sep}$ such that

$$y_i \boldsymbol{\beta}'_{sep} \mathbf{x}_i^* > 0 \quad \forall i \quad \Rightarrow \quad y_i \boldsymbol{\beta}'_{sep} \frac{\mathbf{x}_i^*}{\|\mathbf{x}_i^*\|} > 0 \quad \forall i \quad \Rightarrow \quad y_i \boldsymbol{\beta}'_{sep} \mathbf{z}_i > 0 \quad \forall i.$$

Let $\epsilon = \min_i \{y_i \boldsymbol{\beta}'_{sep} \mathbf{z}_i\} > 0$ and set $\boldsymbol{\beta}'_{sep} = \boldsymbol{\beta}'_{sep} / \epsilon$. Then, $y_i \boldsymbol{\beta}'_{sep} \mathbf{z}_i = \frac{y_i \boldsymbol{\beta}'_{sep} \mathbf{z}_i}{\min_i \{y_i \boldsymbol{\beta}'_{sep} \mathbf{z}_i\}} \geq 1 \quad \forall i$.

$$\begin{aligned}
\text{(b)} \quad \|\boldsymbol{\beta}_{new} - \boldsymbol{\beta}_{sep}\|^2 &= \|\boldsymbol{\beta}_{old} + y_i \mathbf{z}_i - \boldsymbol{\beta}_{sep}\|^2 \\
&= \|\boldsymbol{\beta}_{old} - \boldsymbol{\beta}_{sep}\|^2 + \|y_i \mathbf{z}_i\|^2 + 2y_i \boldsymbol{\beta}'_{old} \mathbf{z}_i - 2y_i \boldsymbol{\beta}'_{sep} \mathbf{z}_i \\
&\leq \|\boldsymbol{\beta}_{old} - \boldsymbol{\beta}_{sep}\|^2 + 1 - 2y_i \boldsymbol{\beta}'_{sep} \mathbf{z}_i \quad \text{since } y_i \boldsymbol{\beta}'_{old} \mathbf{z}_i \leq 0 \text{ by misclassification} \\
&\leq \|\boldsymbol{\beta}_{old} - \boldsymbol{\beta}_{sep}\|^2 - 1 \quad \text{by (a)}.
\end{aligned}$$

(5.2) (16 pts.) (a) Suppose $m = 1$. Then, for all i , if $x \notin [\tau_i, \tau_{i+m}] = [\tau_i, \tau_{i+1}]$, $B_{i,m}(x) = B_{i,1}(x) = 0$. So the claim holds for $m = 1$. Suppose the claim holds for $m = j$ for all $i \in \{1, \dots, k + 2M - m\}$. Then $B_{i,j}(x) = 0$ for $x \notin [\tau_i, \tau_{i+j}]$. Suppose $x \notin [\tau_i, \tau_{i+j+1}]$. Then $x \notin [\tau_i, \tau_{i+j}]$ and hence $B_{i,j}(x) = 0$. Also, $x \notin [\tau_{i+1}, \tau_{i+j+1}]$ and hence $B_{i+1,j}(x) = 0$. Therefore,

$$B_{i,j+1} = \frac{x - \tau_i}{\tau_{i+j} - \tau_i} B_{i,j}(x) + \frac{\tau_{i+j+1} - x}{\tau_{i+j+1} - \tau_{i+1}} B_{i+1,j}(x) = 0$$

when $x \notin [\tau_i, \tau_{i+j+1}]$ and the claim holds for $m = j + 1$. This completes the proof by induction.

(b) Suppose $m = 1$. Then, for all i , if $x \in (\tau_i, \tau_{i+m}) = (\tau_i, \tau_{i+1})$, $B_{i,1} = 1 > 0$. So the claim holds for $m = 1$. Suppose the claim holds for $m = j$, for all $i \in \{1, \dots, k + 2M - m\}$. Then $B_{i,j}(x) > 0$ for $x \in (\tau_i, \tau_{i+j})$. Suppose now that $x \in (\tau_i, \tau_{i+j+1})$. If $x \in (\tau_i, \tau_{i+1}) \subset (\tau_i, \tau_{i+j})$, $B_{i,j}(x) > 0$ and $x \notin [\tau_{i+1}, \tau_{i+j+1}]$, so $B_{i+1,j}(x) = 0$ by (a). If $x \in [\tau_{i+1}, \tau_{i+j}]$, $B_{i,j}(x) > 0$ and $B_{i,j+1}(x) > 0$. If $x \in [\tau_{i+j}, \tau_{i+j+1}) \subset (\tau_{i+1}, \tau_{i+j+1})$, $B_{i,j+1}(x) > 0$ and $x \notin [\tau_i, \tau_{i+1}]$, so $B_{i,j}(x) = 0$ by (a). Thus,

$$B_{i,j+1} = \frac{x - \tau_i}{\tau_{i+j} - \tau_i} B_{i,j}(x) + \frac{\tau_{i+j+1} - x}{\tau_{i+j+1} - \tau_{i+1}} B_{i+1,j}(x) > 0$$

when $x \in (\tau_i, \tau_{i+j+1})$ and the claim holds for $m = j + 1$. This completes the proof by induction.

(c) For $m = 1$, let $x \in [\xi_0, \xi_{k+1}]$. Then $x \in [\tau_m, \tau_{k+m+1}] = [\tau_1, \tau_{k+2}]$ and

$$\sum_{i=1}^{k+1} B_{i,1}(x) = \sum_{i=1}^{k+1} I[\tau_i \leq x < \tau_{i+1}] = 1.$$

Suppose the claim holds for $m = j$. Then $\sum_{i=1}^{k+1} B_{i,j}(x) = 1 \forall x \in [\xi_0, \xi_{k+1}] = [\tau_j, \tau_{j+k+1}]$. Now let $x \in [\xi_0, \xi_{k+1}] = [\tau_{j+1}, \tau_{j+k+2}]$.

$$\begin{aligned}
\sum_{i=1}^{k+j+1} B_{i,j+1}(x) &= \sum_{i=1}^{k+j+1} \left(\frac{x - \tau_i}{\tau_{i+j} - \tau_i} B_{i,j}(x) + \frac{\tau_{i+j+1} - x}{\tau_{i+j+1} - \tau_{i+1}} B_{i+1,j}(x) \right) \\
&= \frac{x - \tau_1}{\tau_{j+1} - \tau_1} B_{1,j}(x) + \sum_{i=2}^{k+j+1} \frac{x - \tau_i}{\tau_{i+j} - \tau_i} B_{i,j}(x) + \sum_{i=1}^{k+j+2} \frac{\tau_{i+j} - x}{\tau_{i+j} - \tau_i} B_{i,j}(x) \\
&= \frac{x - \tau_1}{\tau_{j+1} - \tau_1} B_{1,j}(x) + \sum_{i=2}^{k+j+1} \left(\frac{x - \tau_i}{\tau_{i+j} - \tau_i} + \frac{\tau_{i+j} - x}{\tau_{i+j} - \tau_i} \right) B_{i,j}(x) + \frac{\tau_{k+2j+2} - x}{\tau_{k+2j+2} - \tau_{k+j+2}} B_{k+j+2,j}(x) \\
&= \sum_{i=2}^{k+j+1} B_{i,j}(x) \quad \text{by (a)}.
\end{aligned}$$

Notice that when $m = j$ became $m = j + 1$, the subscripts on the τ_i 's inside $[\xi_0, \xi_{k+1}]$ increased by 1 and so did the $B_{i,m}$'s. So, $\sum_{i=2}^{k+j+1} B_{i,j}(x) = 1$.

(d) For $m = 1$, $B_{i,1}(x) = I[\tau_i \leq x < \tau_{i+1}]$ is a piecewise polynomial of degree 0 with breaks at τ_i and τ_{i+1} . Suppose the results holds for $m = j$. Since

$$B_{i,j+1} = \frac{x - \tau_i}{\tau_{i+j} - \tau_i} B_{i,j}(x) + \frac{\tau_{i+j+1} - x}{\tau_{i+j+1} - \tau_{i+1}} B_{i+1,j}(x)$$

and $B_{i,j}(x)$ and $B_{i+1,j}(x)$ are piecewise polynomials of degree $j - 1$, we have that $B_{i,j+1}(x)$ is a piecewise polynomial of degree j with breaks only at the knots.

(e) not graded

(5.7) (10 pts.) (a) Since g is a natural cubic spline interpolant for $\{x_i, z_i\}_1^N$, it is linear outside $[x_i, x_N]$ and $g''(a) = g''(b) = 0$. Also, $g'''(x)$ is constant on the intervals $[x_i, x_{i+1})$, $i = 1, 2, \dots, N-1$. Since g and \tilde{g} are both functions on $[a, b]$ that interpolate the N pairs, $g(x_i) = \tilde{g}(x_i) = z_i$ and $h(x_i) = 0$ for $i = 1, \dots, N$. Thus,

$$\begin{aligned} \int_a^b g''(x)h''(x)dx &= g''(x)h'(x)|_a^b - \int_a^b g'''(x)h'(x)dx \\ &= - \int_a^b g'''(x)h'(x)dx \\ &= - \sum_{j=1}^{N-1} g'''(x_j^+) \{h(x_{j+1}) - h(x_j)\} \\ &= 0 \end{aligned}$$

(b)

$$\begin{aligned} \int_a^b \tilde{g}''(t)^2 dt &= \int_a^b (h''(t) + g''(t))^2 dt \\ &= \int_a^b h''(t)^2 + \int_a^b g''(t)^2 dt + 2 \int_a^b g''(t)h''(t) dt \\ &= \int_a^b h''(t)^2 + \int_a^b g''(t)^2 dt \quad \text{by (a)} \\ &\geq \int_a^b g''(t)^2 dt \quad \text{with equality holding only if } h \text{ is 0 in } [a, b]. \end{aligned}$$

(c) Since any interpolant f evaluated at x_i yields the same values as $g(x_i)$, the sum of squares $\sum_{i=1}^N (y_i - f(x_i))^2$ will be the same for any choice of f . Thus, since λ is positive, we wish to minimize $\int_a^b f''(t)^2 dt$. In (b) it was shown that this is accomplished by a cubic spline with knots at each of the x_i .

(5.11) (5 pts.) Since $h_1(x) = 1$ and $h_2(x) = x$, we can write $\mathbf{H} = [\mathbf{1}_{N \times 1} \quad \mathbf{x}_{N \times 1} \quad \mathbf{T}_{N \times (N-2)}]$ for an $N \times (N - 2)$ matrix \mathbf{T} . Since $h_1''(x) = 0$, $h_2''(x) = 0$, and $\mathbf{\Omega} = \left(\int h_j''(t)h_k''(t)dt \right)$, the first two

rows and first two columns of $\mathbf{\Omega}$ are $\mathbf{0}$. So, we can write $\mathbf{\Omega} = \begin{bmatrix} \mathbf{0}_{2 \times 2} & \mathbf{0}_{2 \times (N-2)} \\ \mathbf{0}_{(N-2) \times 2} & \mathbf{M}_{(N-2) \times (N-2)} \end{bmatrix}$ for an $(N-2) \times (N-2)$ matrix \mathbf{M} . Now, if we partition \mathbf{H}^{-1} into $\begin{bmatrix} \mathbf{u}'_{1 \times N} \\ \mathbf{v}'_{1 \times N} \\ \mathbf{W}_{(N-2) \times N} \end{bmatrix}$, we have

$$\mathbf{H}^{-1}\mathbf{H} = \begin{bmatrix} \mathbf{u}'\mathbf{1} & \mathbf{u}'\mathbf{x} & \mathbf{u}'\mathbf{T} \\ \mathbf{v}'\mathbf{1} & \mathbf{v}'\mathbf{x} & \mathbf{v}'\mathbf{T} \\ \mathbf{W}\mathbf{1} & \mathbf{W}\mathbf{x} & \mathbf{W}\mathbf{T} \end{bmatrix} = \mathbf{I}_{3 \times 3}.$$

Thus, $\mathbf{W}\mathbf{1} = \mathbf{0}$ and $\mathbf{W}\mathbf{x} = \mathbf{0}$. Let a and b be any constants. Then

$$\mathbf{H}^{-1}(a\mathbf{1} + b\mathbf{x}) = \begin{bmatrix} \mathbf{u}' \\ \mathbf{v}' \\ \mathbf{W} \end{bmatrix} (a\mathbf{1} + b\mathbf{x}) = \begin{bmatrix} \mathbf{u}'(a\mathbf{1} + b\mathbf{x}) \\ \mathbf{v}'(a\mathbf{1} + b\mathbf{x}) \\ \mathbf{0} \end{bmatrix}.$$

Thus, $\mathbf{K}(a\mathbf{1} + b\mathbf{x}) = \mathbf{H}'^{-1}\mathbf{\Omega}\mathbf{H}^{-1}(a\mathbf{1} + b\mathbf{x}) = \mathbf{H}'^{-1} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{u}'(a\mathbf{1} + b\mathbf{x}) \\ \mathbf{v}'(a\mathbf{1} + b\mathbf{x}) \\ \mathbf{0} \end{bmatrix} = \mathbf{0}$, and $\mathbf{1}$ and \mathbf{x} are basis vectors for the null space of \mathbf{K} .

(5.12) (5 pts.) As in Eqn. (5.10) we write $f(x) = \sum_{j=1}^N h_j(x)\theta_j$, where the $h_j(x)$ are an N -dimensional set of basis functions for representing this family of natural splines. The criterion reduces to

$$RSS(\boldsymbol{\theta}, \lambda) = (\mathbf{y} - \mathbf{H}\boldsymbol{\theta})'\mathbf{W}(\mathbf{y} - \mathbf{H}\boldsymbol{\theta}) + \lambda\boldsymbol{\theta}'\mathbf{\Omega}\boldsymbol{\theta},$$

where $\mathbf{W} = \text{diag}(w_1, \dots, w_N)$, and is minimized by

$$\hat{\boldsymbol{\theta}} = (\mathbf{H}'\mathbf{W}\mathbf{H} + \lambda\mathbf{\Omega})^{-1}\mathbf{H}'\mathbf{W}\mathbf{y}.$$

The fitted smoothing spline is given by $\hat{f}(x) = \sum_{j=1}^N h_j(x)\hat{\theta}_j$.

When the training data have ties in \mathbf{X} we remove all ties except one, replacing the y with the average of all y 's for the ties. The technique above can then be used by letting the weight equal the number of ties.

(5.15) (16 pts.) (a)

$$\begin{aligned} \langle K(\cdot, x_i), f \rangle_{\mathcal{H}_K} &= \left\langle \sum_{j=1}^{\infty} (\gamma_j \phi_j(x_i)) \phi_j(\cdot), \sum_{j=1}^{\infty} c_j \phi_j(\cdot) \right\rangle_{\mathcal{H}_K} \\ &= \sum_{j=1}^{\infty} \frac{\gamma_j \phi_j(x_i) c_j}{\gamma_j} = \sum_{j=1}^{\infty} \phi_j(x_i) c_j = f(x_i) \end{aligned}$$

(b)

$$\begin{aligned} \langle K(\cdot, x_i), K(\cdot, x_j) \rangle_{\mathcal{H}_K} &= \left\langle \sum_{k=1}^{\infty} (\gamma_k \phi_k(x_i)) \phi_k(\cdot), \sum_{k=1}^{\infty} (\gamma_k \phi_k(x_j)) \phi_k(\cdot) \right\rangle_{\mathcal{H}_K} \\ &= \sum_{k=1}^{\infty} \frac{\gamma_k \phi_k(x_i) \gamma_k \phi_k(x_j)}{\gamma_k} = \sum_{k=1}^{\infty} \gamma_k \phi_k(x_i) \phi_k(x_j) = K(x_i, x_j) \end{aligned}$$

(c) First,

$$\begin{aligned}
g(x) &= \sum_{i=1}^N \alpha_i K(x, x_i) = \sum_{i=1}^N \alpha_i \sum_{j=1}^{\infty} \gamma_j \phi_j(x) \phi_j(x_i) \\
&= \sum_{j=1}^{\infty} \gamma_j \sum_{i=1}^N (\alpha_i \phi_j(x_i)) \phi_j(x) = \sum_{j=1}^{\infty} c_j \phi_j(x)
\end{aligned}$$

for $c_j = \gamma_j \sum_{i=1}^N \alpha_i \phi_j(x_i)$. Then,

$$\begin{aligned}
J(g) &= \|g\|_{\mathcal{H}_K}^2 = \sum_{j=1}^{\infty} \frac{\gamma_j^2 \left(\sum_{i=1}^N \alpha_i \phi_j(x_i) \right)^2}{\gamma_j} = \sum_{j=1}^{\infty} \gamma_j \left(\sum_{i=1}^N \alpha_i \phi_j(x_i) \right)^2 \\
&= \sum_{j=1}^{\infty} \gamma_j \sum_{i=1}^N \sum_{k=1}^N \alpha_i \alpha_k \phi_j(x_i) \phi_j(x_k) = \sum_{i=1}^N \sum_{k=1}^N \alpha_i \alpha_k \sum_{j=1}^{\infty} \gamma_j \phi_j(x_i) \phi_j(x_k) \\
&= \sum_{i=1}^N \sum_{k=1}^N \alpha_i \alpha_k K(x_i, x_k)
\end{aligned}$$

(d) First,

$$\begin{aligned}
J(\tilde{g}) &= \|g + \rho\|_{\mathcal{H}_K}^2 = \langle g, g \rangle_{\mathcal{H}_K} + 2 \langle g, \rho \rangle_{\mathcal{H}_K} + \langle \rho, \rho \rangle_{\mathcal{H}_K} \\
&= J(g) + 2 \sum_{i=1}^N \alpha_i \langle K(x, x_i), \rho \rangle_{\mathcal{H}_K} + J(\rho) \\
&= J(g) + J(\rho) \geq J(g) \quad \text{with equality holding iff } \rho(x) = 0.
\end{aligned}$$

Now, by (a),

$$\begin{aligned}
\tilde{g}(x_i) &= \langle K(\cdot, x), \tilde{g} \rangle_{\mathcal{H}_K} = \langle K(\cdot, x), g \rangle_{\mathcal{H}_K} + \langle K(\cdot, x), \rho \rangle_{\mathcal{H}_K} \\
&= \langle K(\cdot, x), g \rangle_{\mathcal{H}_K} = g(x_i) \quad \text{for } i = 1, \dots, N.
\end{aligned}$$

Thus, $\sum_{i=1}^N L(y_i, \tilde{g}(x_i)) = \sum_{i=1}^N L(y_i, g(x_i))$, and

$$\sum_{i=1}^N L(y_i, \tilde{g}(x_i)) + \lambda J(\tilde{g}) \geq \sum_{i=1}^N L(y_i, g(x_i)) + \lambda J(g),$$

with equality holding iff $\rho(x) = 0$.

Chapter 4 computing assignment code (20 pts.)

```
##Part 1, according to the outline
vowel=read.table("http://www-stat.stanford.edu/~tibs/ElemStatLearn/datasets/vowel.train",
sep=" ",row.names=1,header=TRUE)

Sig=matrix(rep(0,100),nrow=10)
muk=matrix(nrow=11,ncol=10)
for(i in 1:11){
vowelk=vowel[vowel[,1]==i,2:11]
muk[i,]=apply(vowelk,2,mean)
Sig=Sig+(1/11)*cov(vowelk)}

Siginv=solve(Sig)
e=eigen(Siginv)
V=e$vectors
Sig.half=V%*%diag(sqrt(e$values))%*%t(V)

mubar=apply(muk,2,mean)
mukstar=Sig.half%*%(t(muk)-mubar)
X2=vowel[,2:11]
Xstar=Sig.half%*%(t(X2)-mubar)

W=cov(t(mukstar))
eW=eigen(W)
v=eW$vectors

ok=t(v)
Ox=t(ok%*%Xstar)
Omu=t(ok%*%mukstar)

color=c("black","orange","green","brown","cyan","deeppink","yellow","gray",
"red","darkviolet","blue")
plot(-Ox[,1],Ox[,2],col=rep(color,48),xlab="Coordinate 1 for Training Data",
ylab="Coordinate 2 for Training Data",main="Linear Discriminant Analysis")
points(-Omu[,1],Omu[,2],col=color,pch=19,cex=1.8)

##Parts 2 and 3
library(MASS)
LDA=lda(y~x.1+x.2+x.3+x.4+x.5+x.6+x.7+x.8+x.9+x.10,vowel)
Ox1=predict(LDA,vowel)$x
Omul=predict(LDA, newdata=as.data.frame(LDA$means))$x

par(mfrow=c(2,2))
plot(Ox1[,1],Ox1[,3],col=rep(color,48),xlab="Coordinate 1",ylab="Coordinate 3")
points(Omul[,1],Omul[,3],col=color,pch=19,cex=1.8)
plot(Ox1[,2],Ox1[,3],col=rep(color,48),xlab="Coordinate 2",ylab="Coordinate 3")
points(Omul[,2],Omul[,3],col=color,pch=19,cex=1.8)
plot(Ox1[,1],Ox1[,7],col=rep(color,48),xlab="Coordinate 1",ylab="Coordinate 7")
points(Omul[,1],Omul[,7],col=color,pch=19,cex=1.8)
plot(Ox1[,9],Ox1[,10],col=rep(color,48),xlab="Coordinate 9",ylab="Coordinate 10")
points(Omul[,9],Omul[,10],col=color,pch=19,cex=1.8)
```

```

##Part 4
D=1000
Cols=matrix(nrow=D,ncol=D)
C=matrix(c(-0mu[,1],0mu[,2]),nrow=11)
t1=seq(-5,5,,D)
t2=seq(-7,4,,D)
for(i in 1:D){
for(j in 1:D){
F=((C[,1]-t1[i])^2+(C[,2]-t2[j])^2)
Cols[i,j]=which.min(F)}}

contour(t1,t2,Cols,drawlabels=FALSE,xlab="Coordinate 1",ylab="Coordinate 2",
main="Classified to nearest mean")
points(-0x[,1],0x[,2],col=rep(color,48))
points(-0mu[,1],0mu[,2],col=color,pch=19,cex=1.8)

##Part 5, Logistic Regression
library(VGAM)
logregdata=cbind(rep(1:11,48),-0x[,1],0x[,2])
colnames(logregdata)=c("y","c.1","c.2")
logregdata=as.data.frame(logregdata)
lr=vglm(y~c.1+c.2,family=multinomial(),data=logregdata)

D=500
t1=seq(-5,5,,D)
t2=seq(-7,4,,D)
Col2=matrix(nrow=D,ncol=D)
for(i in 1:D){
Tmat=matrix(c(rep(t1[i],D),t2),ncol=2,nrow=D)
colnames(Tmat)=c("c.1","c.2")
Tmat=as.data.frame(Tmat)
Plr=predict(lr,newdata=Tmat,type="response")
for(m in 1:D){
Col2[i,m]=which.max(Plr[m,])}}

contour(t1,t2,Col2,drawlabels=FALSE,xlab="Coordinate 1",ylab="Coordinate 2",
main="Classified by Logistic Regression")
points(-0x[,1],0x[,2],col=rep(color,48))
points(-0mu[,1],0mu[,2],col=color,pch=19,cex=1.8)

```