Rm 329 Statistics 315a.
in sequoia HomeWork 3. Doug-Bang Tsai

1)

(a) For least squares projections and cubic smoothing splines,

$$\hat{f} = X(X^TX+\lambda\Omega)^{-1}X^Ty. \quad \text{define } A=(X^TX+\lambda\Omega)^{-1}$$

$$\Rightarrow \hat{f} = XA^{-1}X^Ty$$

$$\therefore S = XA^{-1}X^T$$

$$\Rightarrow \hat{f}(x_i) = x_i^T(X^TX+\lambda\Omega)^{-1}X^Ty = x_i^TA^{-1}X^Ty., \quad A = X^TX+\lambda\Omega$$

now $\hat{f}^{-i}(x_i) = x_i^T(X_{-i}^TX_{-i}+\lambda\Omega)^{-1}X_{-i}^Ty_{-i}$

where X_{-i} is all the training set but with x_i removed.

$$\therefore \hat{f}^{-i}(x_i) = x_i^T(X^TX-x_ix_i^T+\lambda\Omega)^{-1}(X^Ty-x_iy_i)$$

$$= x_i^T(A-x_ix_i^T)^{-1}(X^Ty-x_iy_i)$$

Using Mathematica, we can write $(A-x_ix_i^T)^{-1} = A^{-1}+\frac{A^{-1}x_ix_i^TA^{-1}}{(-x_i^TA^{-1}x_i}$

I'll prove it later as well.

$$\hat{f}^{-1}(x_i) = X_i^T \left(A^{-1} + \frac{A^{-1} x_i x_i^T A^{-1}}{1 - x_i^T A^{-1} x_i} \right) (X^T y - x_i y_i)$$

Using $x_i^T A^{-1} x_i = s_{ii}$

$$= \left[x_i^T A^{-1} + \frac{(x_i^T A^{-1} x_i) x_i^T A^{-1}}{1 - (x_i^T A^{-1} x_i)} \right] (X^T y - x_i y_i)$$

$$= \left[x_i^T A^{-1} + \frac{s_{ii} x_i^T A^{-1}}{1 - s_{ii}} \right] (X^T y - x_i y_i)$$

$$= x_i^T A^{-1} X^T y - x_i^T A^{-1} x_i y_i + \left(\frac{s_{ii} x_i^T A^{-1}}{1 - s_{ii}} \right) X^T y - \left(\frac{s_{ii} x_i^T A^{-1}}{1 - s_{ii}} \right) x_i y_i$$

$$= \hat{f}(x_i) - s_{ii} y_i + \frac{s_{ii} \hat{f}(x_i)}{1 - s_{ii}} - \frac{y_i s_{ii}^2}{1 - s_{ii}}$$

$$= \frac{\hat{f}(x_i) - s_{ii} y_i + s_{ii}^2 y_i - y_i s_{ii}^2}{1 - s_{ii}} = \frac{\hat{f}(x_i) - s_{ii} y_i}{1 - s_{ii}}$$

$$= \frac{\hat{f}(x_i) - y_i}{1 - s_{ii}} + y_i$$

$$\Rightarrow y_i - \hat{f}^{-1}(x_i) = \frac{y_i - \hat{f}(x_i)}{1 - s_{ii}}$$ Ans.

Note that

$$(A - x_i x_i^T)\left(A^{-1} + \frac{A^{-1} x_i x_i^T A^{-1}}{1 - x_i^T A^{-1} x_i} \right) = I \quad \therefore (A - x_i x_i^T)^{-1} = A^{-1} + \frac{A^{-1} x_i x_i^T A^{-1}}{1 - x_i^T A^{-1} x_i}$$

(b)

Since $(X_{-i}^T X_{-i} + \lambda\Omega)^{-1}$ is positive-semidefinite,

$$\therefore x_i^T (X_{-i}^T X_{-i} + \lambda\Omega)^{-1} x_i = \left(x_i^T A^{-1} + \frac{S_{ii} x_i^T A^{-1}}{1 - S_{ii}} \right) x_i$$

$$= x_i^T A^{-1} x_i + \frac{S_{ii} x_i^T A^{-1} x_i}{1 - S_{ii}}$$

$$= S_{ii} + \frac{S_{ii}}{1 - S_{ii}} \geq 0$$

$$\Rightarrow 0 \leq S_{ii} < 1$$

$$\therefore \left| y_i - \hat{f}^{-i}(x_i) \right| = \left| \frac{y_i - \hat{f}(x_i)}{1 - S_{ii}} \right| \geq \left| y_i - \hat{f}(x_i) \right|$$

(c) If S depends only on the x_i, and produce $\hat{f}$ from y doesn't depend on y itself.

$$\Rightarrow \hat{f}^{-i}(x_i) = \sum_{j \neq i} S_{ij} y_j + S_{ii} \hat{f}^{-i}(x_i)$$

Since $\hat{f}(x_i) = \sum_j S_{ij} y_j$.

$$\Rightarrow \hat{f}(x_i) - \hat{f}^{-i}(x_i) = S_{ii} \left(y_i - \hat{f}^{-i}(x_i) \right)$$

$\therefore\ y_i - \hat{f}(x_i) = (1-S_{ii})\, y_i - (1-S_{ii})\, \hat{f}^{-1}(x_i)$.

$\Rightarrow\ y_i - f^{-1}(x_i) = \dfrac{y_i - \hat{f}(x_i)}{1 - S_{ii}}$

We get the same result.

2).

ESL 18.8

a) As state in ESL 18.7, we have a linear regression problem where $p >> N$. There are infinitely many least-squares solutions all with zero residuals.

Now, the problem here is we use

$f(x) = \alpha + \beta^T x$ to a classify input x into $+1$ and -1,

$\therefore\ \hat{Y} = X\hat{\beta}$, we know $\hat{\beta}$ in R^p, $\therefore$

Using the result from ESL 18.7,

we can conclude that there are infinitely many directions defind by β in R^p onto which data project to exactly two points.

(b)

The distance of the projected points to the origin is

$$\frac{\hat{\beta}^T x}{\|\hat{\beta}\|}$$

$\therefore$ the distance between the projected points is $\frac{1}{\|\hat{\beta}\|} - \frac{-1}{\|\hat{\beta}\|} = \frac{2}{\|\hat{\beta}\|}$.

(c)

From 18.7 part (c), when $\lambda = 0$, the solution $\hat{\beta}_0 = VD^{-1}U^T y$ has residuals all equal to zero, and is unique in it has the smallest Euclidean norm. However, it means that this $\hat{\beta}_0$ will maximize the distance between the projected points defined by $\frac{2}{\|\hat{\beta}\|}$.

$\therefore$ the single maximal data piling direction can be given by $\hat{\beta}_0 = VD^{-1}U^T y = X^{-1}y$.

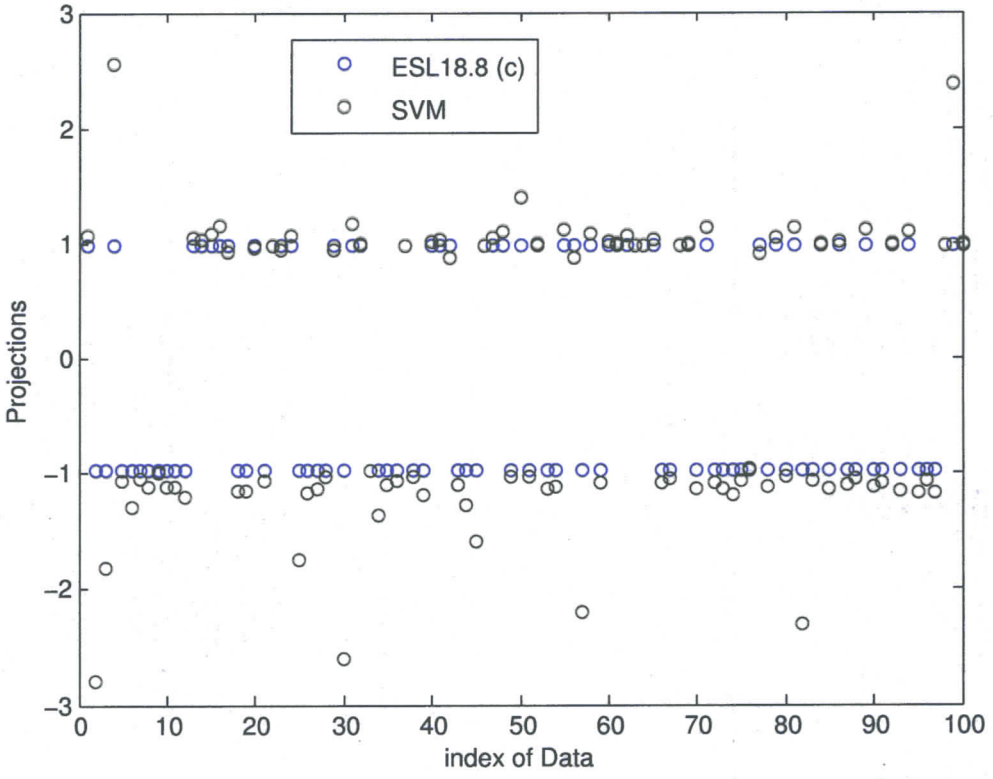
ESL 18.9.

We want to compare the direction of the optimal separating hyperplane to ESL 18.8 (c) here.

Actually, I'll do a small experiment here, $N=20$, $p=100$, randomly generate the sample and map y to -1 and 1.

We'll use 18.8 (c) to obtain the maximal data piling directions. and compare to this one obtained by solving optimal separating hyperplane using SVM.

We will find that the one obtained by SVM will have wider margin than maximal data piling method from ESL 18.8 (c). The code and figure are in the following pages.


ESL18.8 (c)
SVM
Projections
index of Data
3
2
1
0
−1
−2
−3
0
10
20
30
40
50
60
70
80
90
100

```
% HW3 Q2, ESL 18.9
% Dong-Bang Tsai
clear;
N = 100;
p = 200;
X = randn(N, p);
Y = randn(N,1);
for i=1:N
   if Y(i) > 0
       Y(i) = 1;
   else
       Y(i) = -1;
   end
end
[U,S,V] = svd(X, 'econ');
beta_p = (V/S)*U'*Y;
% The following is the projection distance obtained by ESL18.8 (c)
D_p = X*beta_p/sqrt(sum(beta_p.^2));

SVMStruct = svmtrain(X,Y);
Group = svmclassify(SVMStruct,X);
beta_s = SVMStruct.SupportVectors'*SVMStruct.Alpha;
% The following is the projection distance obtained by SVM
D_s = -X*beta_s/sqrt(sum(beta_s.^2));

x_indx = [1:N];
plot(x_indx, D_p, 'bo', x_indx, D_s, 'ro');
xlabel('index of Data'); ylabel('Projections');
legend('ESL18.8 (c)','SVM')
```

3). ESL 5.5

The source code is in the next page.

For the raw data, there is no training error.

and the testing error will be 0.1583

Since in the raw data, we have 256 degree of freedom in each sample. therefor, in order to have 5 different choice for the nodes, we use

knots = [4, 8, 16, 32, 64].

We find out that coressponding error will be

[0.0955, 0.0826, 0.0826, 0.0775, 0.0892]

∴ We pick 32 knots in our case.

⇒ Full traing error : 0.0530

Full testing error : 0.0831

using cubic spline.

	Raw	filtered
Traing error	0	0.0530
Testing error	0.1583	0.0831

As we expect, the test error decrease but training error increase when we apply smooth.

```
% HW3 Q3, ESL 18.9
% Dong-Bang Tsai
clear;

% type of phonemes, 1)aa  2)ao 3)dcl 4)iy 5)sh
N_sample = 4509;
Y=zeros(N_sample,1);
Y_type = -1*ones(N_sample,1); % 0 is training set, 1 is testing set.
X=zeros(N_sample,256);

% This part of reading data takes most of mine time.
scanformate = '%d';
for i=1:256
    scanformate = strcat(scanformate, ', %f');
end
scanformate = strcat(scanformate, ', %s');
fid = fopen('phoneme.data');
tline = fgetl(fid);
tline = fgetl(fid);
for i=1:N_sample
    C = textscan(tline,scanformate);
    for j=1:256
        X(i,j) = C{j+1};
    end
    [tok,rem] = strtok(C{258},',');
    if strcmp(tok, 'aa')
        Y(i) = 1;
    elseif  strcmp(tok, 'ao')
        Y(i) = 2;
    elseif strcmp(tok,'dcl')
        Y(i) = 3;
    elseif strcmp(tok,'iy')
        Y(i) = 4;
    elseif strcmp(tok,'sh')
        Y(i) = 5;
    end
    [tok,rem] = strtok(rem,',');
    [tok,rem] = strtok(tok,'.');
    if strcmp(tok,'train')
        Y_type(i) = 0;
    elseif strcmp(tok,'test')
        Y_type(i) = 1;
    end
    tline = fgetl(fid);
end
fclose(fid);

indx_train = find(Y_type == 0);
Y_train = Y(indx_train);
X_train = X(indx_train,:);

indx_test = find(Y_type == 1);
Y_test = Y(indx_test);
X_test = X(indx_test,:);

Y_train_est = classify(X_train,X_train, Y_train,'quadratic');
misclassified_rate_train_raw = sum(Y_train_est ~= Y_train)/length(Y_train);

Y_test_est = classify(X_test, X_train, Y_train,'quadratic');
misclassified_rate_test_raw = sum(Y_test_est ~= Y_test)/length(Y_test);
```

```
61
62 knots = [50 100 150 200 250];
63 for i = 1:5
64     sp = spap2(knots(i),4,[1:256],X_train);
65     SpData{i} = sp.coefs;
66 end
67
68 error_knots = zeros(5,1);
69 for i=1:5
70     Y_sp = Y_train;
71     X_sp = SpData{i};
72    for j=1:10
73         Y_sp_test =  Y_sp(334*j +1:334*(j+1));
74         Y_sp_train = Y_sp;
75         %Y_sp_train(334*j +1:334*(j+1)) =[];
76         %X_sp_test = X_sp(334*j +1:334*(j+1),:);
77         X_sp_train = X_sp;
78         %X_sp_train(334*j +1:334*(j+1),:) = [];
79         %Y_sp_test_est = classify(X_sp_test,X_sp_train, Y_sp_train,'quadratic');
80         %error_knots(i) = error_knots(i) + sum(Y_sp_test_est ~= Y_sp_test)/length
   (Y_sp_test);
81    end
82 end
83
84
85
```

4) ESL 5.7

Integration by parts will give us

$$\int_a^b g''(x) h''(x)\, dx = \left[h'(x) g''(x) \right]_a^b - \int_a^b g'''(x) h'(x)\, dx$$

(ps: first term is zero, and $g'''(x) = 0$ when $x \le a$ or $x \ge b$)

$$= -\int_{x_1}^{x_n} g'''(x) h'(x)\, dx$$

$$= -\sum_{j=1}^{N-1} \int_{x_K}^{x_{j+1}} g'''(x) h'(x)\, dx$$

Since g is constructed by cubic spline, $\therefore g'''(x) = \text{constant}$ in the interval.

$$\therefore \int_a^b g''(x) h''(x)\, dx = -\sum_{j=1}^{N-1} g'''(x_j^+) \int_{x_K}^{x_{j+1}} h'(x)\, dx$$

$$= -\sum_{j=1}^{N-1} g'''(x_j^+) \left[h(x_{j+1}) - h(x_j) \right]$$

Since it's interpolation, at node, $g(x_j) = \hat{g}(x_j)$

$\therefore h(x_j) = 0$

$$\therefore \int_a^b g''(x) h''(x)\, dx = 0.$$

(b) $\tilde{g} = h(x) + g(x)$

$$\int_a^b \tilde{g}''(t)^2\,dt = \int_a^b (h''(t) + g''(t))^2\,dt.$$

$$= \int_a^b (h''(t))^2 + (g''(t))^2 + 2h''(t)g''(t)\,dt$$

$$= \int_a^b (h''(t))^2 + (g''(t))^2\,dt$$

$$\geq \int_a^b g''(t)^2\,dt.$$

(c) If $\int_a^b h''(t)^2\,dt = 0$, then the equality holds.

The condition will be

$$h''(t) = 0 \quad \forall\, t \in [a, b].$$

(c)

We want to find the function $\hat{f}$ such that

$$\hat{f} = \arg\min_f \left[\sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \int_a^b f''(t)^2 \, dt \right]$$

Now, g is a natural cubic spline with knots at each of the x_i. Suppose that both g and $\hat{f}$ interpolate the same sequence, we can write:

$$\int_a^b \hat{f}''(t)^2 \, dt \geq \int_a^b g''(t)^2 \, dt \quad \text{(from (b))}$$

It imply that $h(t) = \hat{f}(t) - g(t) = 0 \quad \forall t \in [a, b]$

since the loss term and penalty term with g will be never greater than the one with $\hat{f}$.

$$\Rightarrow \hat{f}(t) = g(t).$$

5) ESL 5.16.

From 5.45 of the textbook,

$$K(x,y) = \sum_{m=1}^{M} h_m(x)\, h_m(y) = \sum_{i=1}^{\infty} \gamma_i \phi_i(x) \phi_i(y)$$

Multiply $\phi_j(x)$ on each side, and integrate it.

$$\sum_{m=1}^{M} \left(\int h_m(x) \phi_j(x) dx \right) h_m(y) = \sum_{i=1}^{\infty} \gamma_i \left(\int \phi_i(x) \phi_j(x) dx \right) \phi_i(y)$$

Using the orthogonality of $\phi_j(x)$, ie,

$$\int \phi_i(x) \phi_k(x) dx = \begin{cases} 1, & \text{when } i = k \\ 0 & \text{when } i \neq k \end{cases}$$

$$\therefore \sum_{m=1}^{M} \left(\int h_m(x) \phi_j(x) dx \right) h_m(y) = \gamma_j \phi_j(y)$$

Multiply with $\phi_k(y)$ and integrate again.

$$\sum_{m=1}^{M} \left[\int h_m(x) \phi_j(x) dx \right] \left[\int h_m(y) \cdot \phi_k(y) dy \right] = \gamma_j \int \phi_j(y) \phi_k(y) dy$$

$$= \gamma_j \delta_{j,k}$$ where we apply orthogonality!

Write down this equation in the matrix form.

then

$$\left(\int h(x)\,\phi(x)\,dx\right)\left(\int h(y)\,\phi(y)\,dy\right) = \mathrm{diag}(r_1, r_2, \ldots r_M) = D_r.$$

define

$$V^T = D_r^{-\frac{1}{2}} \int h(x)\,\phi(x)\,dx \quad -(1)$$

$V^TV = VV^T = I$, $\therefore$ V is an orthogonal matrix.

$$\therefore \left(\int h(x)\,\phi(x)\,dx\right) h(y) = D_r\,\phi(y)$$

and $$\sum_{m=1}^{M} \left(\int h_m(x)\,\phi_j(x)\,dx\right) h_m(y) = r_j\,\phi_j(y)$$

with eq (1).

$$D_r^{\frac{1}{2}} V^T h(y) = D_r\,\phi(y)$$

Using V is orthogal, we can write.

$$h(x) = V D_r^{\frac{1}{2}}\,\phi(x)$$

from 5.63

$$\min_{\{\beta_m\}_1^M} \sum_{i=1}^{N} \left(y_i - \sum_{m=1}^{M} \beta_m h_m(x_i) \right)^2 + \lambda \sum_{m=1}^{M} \beta_m^2 .$$

$$= \min_{\{\beta_m\}_1^m} \sum_{i=1}^{N} (y_i - \beta^T h(x_i))^2 + \lambda \beta^2$$

$$= \min \sum_{i=1}^{N} (y_i - \beta^T V D_\gamma^{\frac{1}{2}} \phi(x_i))^2 + \lambda \beta^T \beta \quad \left(\text{define } \beta = V D_\gamma^{-\frac{1}{2}} c \right)$$

$$= \min_c \sum_{i=1}^{N} (y_i - c^T \phi(x_i))^2 + \lambda (V D_\gamma^{-\frac{1}{2}} c)^T V D_\gamma^{-\frac{1}{2}} c$$

$$= \min_c \sum_{i=1}^{N} (y_i - c^T \phi(x_i))^2 + \lambda c^T D_\gamma^{-\frac{1}{2}} \underbrace{V^T V}_{I} D_\gamma^{-\frac{1}{2}} c$$

$$= \min_c \sum_{i=1}^{N} (y_i - c^T \phi(x_i))^2 + \lambda c^T c D_\gamma^{-1}$$

$$= \min_{\{c_j\}_1^\infty} \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{\infty} c_j \phi_j(x_i) \right)^2 + \lambda \sum_{j=1}^{\infty} \frac{c_j^2}{\gamma_j} .$$

(b).

$\therefore \hat{\beta} = (H^T H + \lambda I)^{-1} H^T y.$

$\therefore \hat{f} = H\hat{\beta} = H(H^T H + \lambda I)^{-1} H^T y.$

$= \left(\underbrace{(H^T)^{-1} H^T H H^{-1}}_{I} + \lambda \underbrace{(H^T)^{-1} H^{-1}}_{K} \right)^{-1} y$

$= (I + \lambda K^{-1})^{-1} y.$

$\therefore \hat{f} = K(K + \lambda I)^{-1} y.$

Note that $M < N$

$\therefore$ $K + \lambda I$ are invertible

(c).

Let $\hat{\alpha} = (K + \lambda I)^{-1} y.$

$\hat{f} = K\hat{\alpha}$. from part (b).

$\therefore f(x) = h(x)^T \hat{\beta} = \sum_{i=1}^{N} K(x, x_i) \hat{\alpha}_i.$

(d)

If $M < N$.

K is not invertible, but $\forall \lambda \neq 0$, $K + \lambda I$ is invertible. ∴ The result still works

$$\hat{f} = H\hat{\beta} = K(K + \lambda I)^{-1} y$$

when $\lambda = 0$,

$$\hat{f} = H\hat{\beta} = H(H^T H)^{-1} H^T y = y$$

#1.