Stats 315A – HW1 Solutions

6th February, 2012

If there are any questions regarding the solutions or the grades of HW 1, please contact Gourab (gourab@stanford.edu) with 'Stats315A-hw1-grading' in the subject line. Common mistakes are highlighted in blue.

Grade Distribution: Total 100 Points

Problem 1: 30 [5+ 10+ 10 + 5]
Problem 2: 10
Problem 3: 15 [3+5+4+3]
Problem 4: 10
Problem 5: 20 [10+10]
Problem 6: 15 [4+3+4+4]

Problem 1

Part (a):

The code is provided at the end of Problem 1.
1 Point has been deducted for codes containing long 'for loops'.

Part (b):

The misclassification errors (both training and test) are evaluted for K-NN at $k = \{1; 3; 5; 9; 15; 25; 45; 83; 151\}$. The plot is shown in Figure 1.

Part (c):

10 fold cross validation is used to show how the test errors (estimated via cross-validation, averaging out over the 10 folds) vary with varying # of neighbors for the KNN procedure. The results are shown in Figure 2.

Part (d):

In Figure 1, the training errors decrease with decrease in # neighbors, however the test error becomes quite large for smaller values of k due to an increase in the variance of the estimator. The test errors are always above the training errors. Linear regression with df = 3 tends to perform similar to the best KNN.

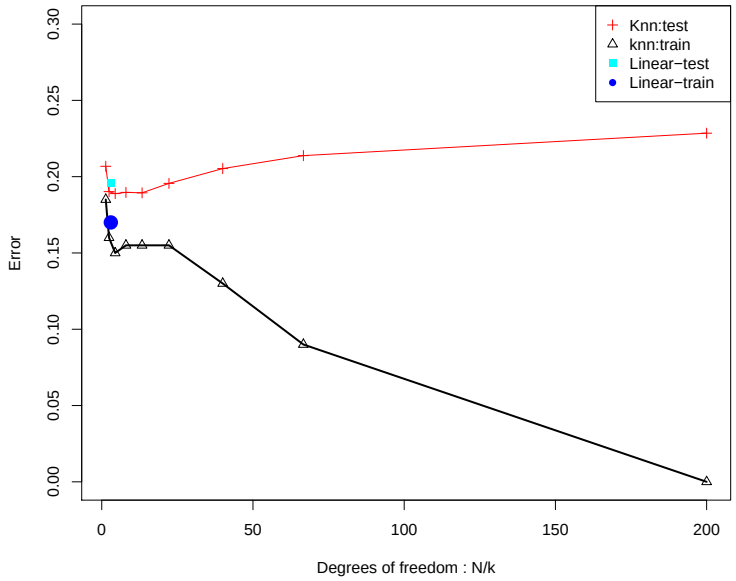

Figure 1: Figure showing the classification errors for KNN procedure plotted vs degrees of freedom. Both the training and test errors are shown. The figure also displays the results for linear regression.

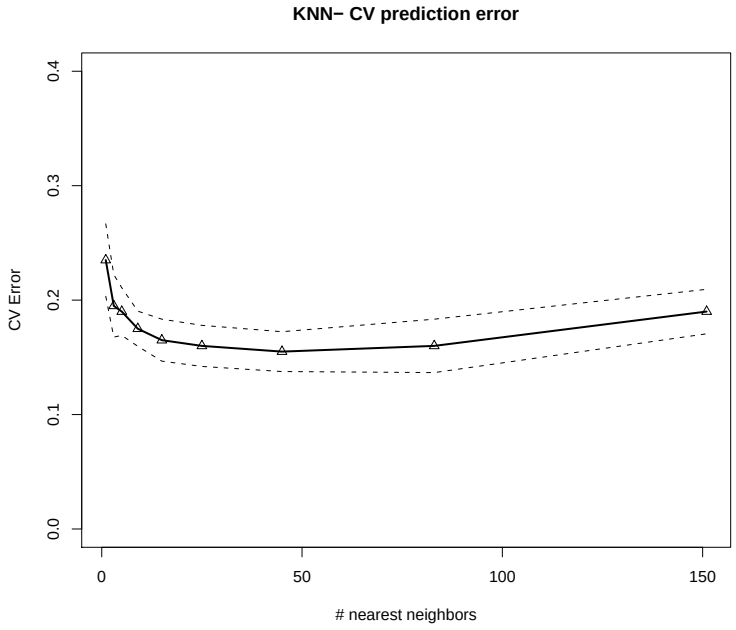

Figure 2: Figure showing the classification errors, along with the standard error bands (dotted). The CV errors provide estimates of the prediction accuracy of different KNN procedures with different K values. They can also be used to choose an optimal value of the tuning parameter ie the number of neighbors in KNN.

Th CV error curves are a proxy for prediction errors (ie classification errors). The CV averaged fold errors suggest a minima at $k = 45$. However the error bands suggest that the estimate of the CV error is quite noisy, so there is quite some variability in the estimated CV errors.

R code for Problem 1:

```
rm(list=ls())

## function to generate mixture gaussians fro two classes
set.seed(200)

##Part a)

generate.mixture<- function(centroids,sample.size=100,noise.var=1) {
no.components<-nrow(centroids);
sigma<-sqrt(noise.var);
ids<-ceiling(runif(sample.size)*no.components);
X=cbind(rnorm(sample.size,centroids[ids,1],sigma),rnorm(sample.size,centroids[ids
    ,2],sigma));
return(X);
}

## generate and store centroids
centroids_class1<-cbind(rnorm(10,mean=1,sd=1),rnorm(10,mean=0,sd=1));
centroids_class0<-cbind(rnorm(10,mean=0,sd=1),rnorm(10,mean=1,sd=1));

## generate mixture

## for training
tr_class1<-generate.mixture(centroids_class1,sample.size=100,noise.var=1/5)
tr_class0<-generate.mixture(centroids_class0,sample.size=100,noise.var=1/5)
Train.x<-rbind(tr_class1,tr_class0);
Train.y<-c(rep(1,100),rep(0,100));

## test data
te_class1<-generate.mixture(centroids_class1,sample.size=5000,noise.var=1/5)
te_class0<-generate.mixture(centroids_class0,sample.size=5000,noise.var=1/5)
Test.x<-rbind(te_class1,te_class0);
Test.y<-c(rep(1,5000),rep(0,5000));

plot(tr_class1[,1],tr_class1[,2],xlim=c(-3,3),ylim=c(-3,3))
points(tr_class0[,1],tr_class0[,2],xlim=c(-3,3),ylim=c(-3,3),col=2)

x11()
plot(te_class1[,1],te_class1[,2],xlim=c(-3,3),ylim=c(-3,3))
points(te_class0[,1],te_class0[,2],xlim=c(-3,3),ylim=c(-3,3),col=2)



## part (b):

## perform KNN

k.vec<-c(1,3,5,9,15,25,45,83,151);
library(class)
```

```
tr_error<-0*k.vec;
te_error<-tr_error;

for (i in 1:length(k.vec)) {
test<-rbind(Train.x,Test.x)
A<-knn(train=Train.x,test=test,cl=Train.y, k=k.vec[i])
tr_error[i]<-mean(A[1:length(Train.y)]!=Train.y);
te_error[i]<-mean(A[-c(1:length(Train.y))]!=Test.y);
}

## Linear regression:

lin.pred<-lm(Train.y ~., data=as.data.frame(Train.x))
lin.reg_tr<-predict.lm(lin.pred,as.data.frame(Train.x))>0.5;
lin.reg_te<-predict.lm(lin.pred,as.data.frame(Test.x))>0.5;
lin.reg_tr.err<-mean(lin.reg_tr!=Train.y);
lin.reg_te.err<-mean(lin.reg_te!=Test.y);

plot(nrow(Train.x)/k.vec,tr_error,ylab="Error",xlab="Degrees of freedom : N/k",type=
    "l",lwd=2,ylim=c(0,0.3))
points(nrow(Train.x)/k.vec,tr_error,col=1,pch=2)
lines(nrow(Train.x)/k.vec,te_error,col=2)
points(nrow(Train.x)/k.vec,te_error,col=2,pch=3)
points(3,lin.reg_tr.err,col=4,type="b",pch=16,cex=2)
points(3,lin.reg_te.err,col=5,type="b",pch=15)
legend("topright",c("Knn:test","knn:train","Linear-test","Linear-train"),pch=c
    (3,2,15,16),col=c(2,1,5,4))



### Part (c): CV

N=nrow(Train.x); K.fold=10;
foldid= sample(rep(1:K.fold, out.length=N))
## returns matrix of leave-out-pred error for a lambda averaged over 10 folds
cv_error.knn<- array(0,dim=c(K.fold,length(k.vec)));

cv.lin<-rep(0,K.fold);

for (kk in 1:length(k.vec)) {
for (fold in 1:K.fold) {
train=(foldid!=fold); test=(foldid==fold);
A<-knn(train=Train.x[train,],test=Train.x[test,],cl=Train.y[train], k=k.vec[kk])
cv_error.knn[fold,kk]<-mean(A!=Train.y[test]);

lin.pred<-lm(Train.y[train] ~., data=as.data.frame(Train.x[train,]))
lin.reg_te<-predict.lm(lin.pred,as.data.frame(Train.x[test,]))>0.5;
cv.lin[fold]<-mean(lin.reg_te!=Train.y[test]);
  }
}

cv_ave.err<-apply(cv_error.knn,2,mean);
cv_sd.err<-apply(cv_error.knn,2,sd)/sqrt(K.fold);
cv_ave.lin<-mean(cv.lin);cv_sd.lin<-sd(cv.lin)/sqrt(K.fold);
```

```
103
x11()
105 plot(k.vec,cv_ave.err,ylab="CV Error",xlab="# nearest neighbors",type="l",lwd=2,ylim
    =c(0.0,0.4),
main="KNN- CV prediction error")
107 points(k.vec,cv_ave.err,col=1,pch=2)
lines(k.vec,cv_ave.err - cv_sd.err,col=1,lty=2)
109 lines(k.vec,cv_ave.err + cv_sd.err,col=1,lty=2)
#legend("topright",c("Knn-Test(CV)","KNN-test"),pch=c(2,1),col=c(2,1))
```

R-CODE

Problem 2 (ESL Exercise 2.4)

Here we have, for $i \neq 0$

$$z_i = \frac{x_0^T}{\|x_0\|} x_i, \text{ where } x_i \overset{i.i.d}{\sim} N(0, \mathbf{I}_p).$$

Conditional on x_0, we get

$$z_i | x_0 = \frac{1}{\|x_0\|} \sum_{k=1}^{p} x_{0,k} x_{i,k} \overset{d}{=} N\left(0, \frac{\sum_{k=1}^{p} x_{0,k}^2}{\|x_0\|^2}\right) = N\left(0, \frac{\|x_0\|^2}{\|x_0\|^2}\right) = N(0,1) \tag{1}$$

as linear combination of normals is also normally distributed.

Therefore, conditionally $z_i | x_0 \sim N(0,1)$, which, however, is independent of x_0. Thus, $z_i \sim N(0,1)$, unconditionally. And so $\mathrm{E}\left(z_i^2\right) = 1$. Also,

$$\mathrm{E}\left(\|x_i\|^2\right) = \mathrm{E}\left(\sum_{k=1}^{p} x_{i,k}^2\right) = \sum_{k=1}^{p} \mathrm{E}\left(x_{i,k}^2\right) = p \text{ for all } i$$

and so we clearly see that the prediction point is away from the projections of the data.

Problem 3 (ESL Exercise 2.7)

(a) For linear regression:

$$\hat{f}(x_0) = x_0^T \hat{\beta} = x_0^T (X^T X)^{-1} X^T Y$$

So $l_i(x_0; \mathcal{X}) = x_0^T (X^T X)^{-1} x_i$.

For K-nearest-neighbor regression:

$$\hat{F}(x_0) = \frac{1}{k} \sum_{x_i \in N_k(x_0)} y_i$$

So $l_i(x_0; \mathcal{X}) = 1/k$ if $x_i \in N_k(x_0)$, 0 otherwise.

(b) Begin with the familiar bias-variance decomposition:

$$\begin{aligned} E_{\mathcal{Y}|\mathcal{X}}(f(x_0) - \hat{f}(x_0))^2 &= E_{\mathcal{Y}|\mathcal{X}}(f(x_0) - E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0))^2 + E_{\mathcal{Y}|\mathcal{X}}(E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0) - \hat{f}(x_0))^2 \\ &= Bias^2_{\mathcal{Y}|\mathcal{X}}(\hat{f}(x_0)) + Var_{\mathcal{Y}|\mathcal{X}}(\hat{f}(x_0)) \end{aligned}$$

Now, we express the bias and variance in terms of l_i.

$$\begin{aligned} Bias_{\mathcal{Y}|\mathcal{X}}(\hat{f}(x_0)) &= E_{\mathcal{Y}|\mathcal{X}}\left(\sum_{i=1}^{N} l_i(x_0;\mathcal{X})y_i - f(x_0)\right) \\ &= \sum_{i=1}^{N} l_i(x_0;\mathcal{X})f(x_i) - f(x_0) \end{aligned}$$

$$\begin{aligned} Var_{\mathcal{Y}|\mathcal{X}}(\hat{f}(x_0)) &= Var_{\mathcal{Y}|\mathcal{X}}\left(\sum_{i=1}^{N} l_i(x_0;\mathcal{X})y_i\right) \\ &= \sigma^2 \sum_{i=1}^{N} l_i^2(x_0;\mathcal{X}) \end{aligned}$$

Some points are deducted for not simplifying these expressions.

(c) As before, the expression decomposes into squared bias and variance terms.

$$E_{\mathcal{X},\mathcal{Y}}(f(x_0) - \hat{f}(x_0))^2 = Bias^2_{\mathcal{X},\mathcal{Y}}(\hat{f}(x_0)) + Var_{\mathcal{X},\mathcal{Y}}(\hat{f}(x_0))$$

Because these terms are now unconditional, we must re-calculate in terms of l_i:

$$\begin{aligned} Bias_{\mathcal{X},\mathcal{Y}}(\hat{f}(x_0)) &= E_{\mathcal{X}} Bias_{\mathcal{Y}|\mathcal{X}}(\hat{f}(x_0)) \\ &= E_{\mathcal{X}} \sum_{i=1}^{N} l_i(x_0;\mathcal{X})f(x_i) - f(x_0) \end{aligned}$$

$$\begin{aligned} Var_{\mathcal{X},\mathcal{Y}}(\hat{f}(x_0)) &= E_{\mathcal{X},\mathcal{Y}}(\hat{f}(x_0) - E_{\mathcal{X},\mathcal{Y}}\hat{f}(x_0))^2 \\ &= E_{\mathcal{X}} E_{\mathcal{Y}|\mathcal{X}}(\hat{f}(x_0) - E_{\mathcal{X},\mathcal{Y}}\hat{f}(x_0))^2 \\ &= E_{\mathcal{X}}\left[E_{\mathcal{Y}|\mathcal{X}}\left(\sum_{i=1}^{N} l_i(x_0;\mathcal{X})y_i - E_{\mathcal{X}}(l_i(x_0;\mathcal{X})f(x_i))\right)^2\right] \\ &= E_{\mathcal{X}}\left[\sigma^2 \sum_{i=1}^{N} l_i^2(x_0;\mathcal{X})\right] + Var_{\mathcal{X}}\left[\sum_{i=1}^{N} l_i(x_0;\mathcal{X})f(x_i)\right]. \end{aligned}$$

(d) For variance in general:

$$V_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0) = V_{\mathcal{X}}\left[E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0)\right] + E_{\mathcal{X}}\left[V_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0)\right]$$

So we can always say:

$$V_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0) \geq E_{\mathcal{X}} V_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0)$$

In other words, the unconditional variance is bigger than the "average" conditional variance by the amount $V_{\mathcal{X}}\left[E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0)\right]$.

For bias:

$$\begin{aligned}
&E_{\mathcal{X}}\big(f(x_0) - E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0)\big)^2 \\
&= E_{\mathcal{X}}\big(f(x_0) - E_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0) + E_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0) - E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0)\big)^2 \\
&= (f(x_0) - E_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0))^2 + E_{\mathcal{X}}(E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0) - E_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0))^2
\end{aligned}$$

So we get the opposite "average" inequality, i.e. the unconditional bias is smaller than the mean conditional bias:

$$(f(x_0) - E_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0))^2 \leq E_{\mathcal{X}}(f(x_0) - E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0))^2$$

And, the difference is $V_{\mathcal{X}}\left[E_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0)\right]$.

It is worth remembering that the above inequalities are only true on average – in general, there exist training samples from $\mathcal{X}$ that will decrease the bias or raise the variance of $\hat{f}(x_0)$. However, K-nearest-neighbor regression is a special case in that $V_{\mathcal{Y}|\mathcal{X}}\hat{f}(x_0) \equiv 1/N \cdot \sigma^2$, thus $V_{\mathcal{Y},\mathcal{X}}\hat{f}(x_0) \geq \sigma^2/N$.

A lot of points have been deducted for wrong interpretation in (d).

Problem 4 (ESL Exercise 2.9)

First note

$$E(R_{te}(\hat{\beta})) \quad = \quad \frac{1}{M}\sum_{i=1}^{M} E((\tilde{y}_i - \hat{\beta}\tilde{x}_i)^2) = \frac{1}{N}\sum_{i=1}^{N} E((\tilde{y}_i - \hat{\beta}\tilde{x}_i)^2)$$

because $\hat{\beta}$ has no dependence on $(\tilde{x}_i, \tilde{y}_i)$ or M. Thus, without loss of generality assume that $M = N$. Then let $\tilde{\beta}$ be the least squares estimate of β based on the test data and by symmetry

$$E(R_{te}(\hat{\beta})) = E(R_{tr}(\tilde{\beta})). \tag{2}$$

By properties of least square estimators we have $R_{tr}(\hat{\beta}) \leq R_{tr}(\tilde{\beta})$. Now, by monotonicity of the expectation operator and by equation 2 we get the desired result.

Many students asserted that

$$E(R_{tr}(\hat{\beta})) = E(R_{te}(\tilde{\beta}))$$

but this is only true when $M = N$ since the usual regression arguments give

$$E(R_{tr}(\hat{\beta})) = \frac{N-p}{N}\sigma^2$$

which depends on N.

Problem 5 (ESL Exercise 3.2)

Both approaches are based on the fact that $\hat{\beta} \sim N(\beta, \sigma^2(\mathbf{X}^T\mathbf{X})^{-1})$ (Equation (3.10)). The conceptual difference between the two methods' confidence intervals (denote them by CI_i^a and CI_{ii}^a) is the following:

$$\begin{aligned} P(a^T\beta \in CI_i^a) \quad &\geq 1-\alpha \qquad \forall a \\ P(a^T\beta \in CI_{ii}^a \ \forall a) &\geq 1-\alpha. \end{aligned}$$

We would expect CI_{ii}^a to be wider since its notion of coverage is simultaneous for *all* a.

Method i

By definition of the multivariate normal distribution, we know that $a^T\hat{\beta}$ is normal. Also,

$$\begin{aligned} \mathbb{E}[a^T\hat{\beta}] &= a^T\beta \\ \mathbb{V}ar[a^T\hat{\beta}] &= a^T\mathbb{C}ov[\hat{\beta}]a = \sigma^2 a^T(\mathbf{X}^T\mathbf{X})^{-1}a. \end{aligned}$$

So $a^T\hat{\beta} \sim N(a^T\beta, \sigma^2 a^T(\mathbf{X}^T\mathbf{X})^{-1}a)$. It follows that

$$P(-z_{1-\alpha/2} \leq \frac{a^T\hat{\beta} - a^T\beta}{\sigma\sqrt{a^T(\mathbf{X}^T\mathbf{X})^{-1}a}} \leq z_{1-\alpha/2}) = 1-\alpha.$$

Thus, a 95% confidence interval at the point a is given by

$$CI_i^a = \left[a^T\hat{\beta} - z_{0.975}\sigma\sqrt{a^T(\mathbf{X}^T\mathbf{X})^{-1}a}, \quad a^T\hat{\beta} + z_{0.975}\sigma\sqrt{a^T(\mathbf{X}^T\mathbf{X})^{-1}a}\right].$$

if σ is known, or else by

$$CI_i^a = \left[a^T\hat{\beta} - t_{n-4}^{0.975}\hat{\sigma}\sqrt{a^T(\mathbf{X}^T\mathbf{X})^{-1}a}, \quad a^T\hat{\beta} + t_{n-4}^{0.975}\hat{\sigma}\sqrt{a^T(\mathbf{X}^T\mathbf{X})^{-1}a}\right].$$

Method ii

In method 2, we begin with the confidence set for the vector $\hat{\beta}$ given in Equation (3.15):

$$C_\beta = \left\{\beta : (\hat{\beta}-\beta)^T\mathbf{X}^T\mathbf{X}(\hat{\beta}-\beta) \leq \sigma^2\chi_4^2(1-0.05)\right\}.$$

This follows because

$$(\hat{\beta}-\beta)^T\mathbf{X}^T\mathbf{X}(\hat{\beta}-\beta) \sim \sigma^2\chi_4^2.$$

If σ^2 is unknown, then technically it should be

$$C_\beta = \left\{\beta : (\hat{\beta}-\beta)^T\mathbf{X}^T\mathbf{X}(\hat{\beta}-\beta) \leq \hat{4}\sigma^2 F_{4,n-4}(1-0.05)\right\}.$$

But, for large n, this makes little difference as the cut-offs from the cumbersome t and F distributions can be well approximated by corresponding quantiles from Z and χ^2 respectively.
Some points are deducted from answers containing intervals based on Z and χ^2 cut-offs and not mentioning the large

sample approximations.

Now, $CI^a_{ii} = \{a^T\beta : \beta \in C_\beta\}$, so

$$0.95 = P(\beta \in C_\beta) \leq P(a^T\beta \in CI^a_{ii}\ \forall a)$$

Geometrically, C_β is an ellipse in p-dimensional space, and CI^a_{ii} is the projection of this ellipse onto the subspace spanned by a. To find the endpoints of the interval CI^a_{ii}, we wish to solve

$$\min_\beta / \max_\beta a^T\beta \text{ subject to } (\hat{\beta}-\beta)^T\mathbf{X}^T\mathbf{X}(\hat{\beta}-\beta) \leq \sigma^2\chi^2_4(0.95).$$

By recentering and sphering, we may transform this to a simpler problem: Let $z = (\mathbf{X}^T\mathbf{X})^{1/2}(\beta - \hat{\beta})$. Then, noting that $\beta = \hat{\beta} + (\mathbf{X}^T\mathbf{X})^{-1/2}z$, we rewrite the problem as

$$\min_z / \max_z a^T(\hat{\beta} + (\mathbf{X}^T\mathbf{X})^{-1/2}z) \text{ subject to } ||z||^2 \leq \sigma^2\chi^2_4(0.95).$$

By Cauchy-Schwarz, this is minimized/maximized at $z = \pm c(\mathbf{X}^T\mathbf{X})^{-1/2}a$, where $c = \frac{\sigma\chi^2_4(0.95)}{||(\mathbf{X}^T\mathbf{X})^{-1/2}a||}$. Therefore, the endpoints are given by

$$\begin{aligned} a^T(\hat{\beta} \pm (\mathbf{X}^T\mathbf{X})^{-1/2}c(\mathbf{X}^T\mathbf{X})^{-1/2}a)) &= a^T\hat{\beta} \pm \frac{\sigma\sqrt{\chi^2_4(0.95)}a^T(\mathbf{X}^T\mathbf{X})^{-1}a}{\sqrt{a^T(\mathbf{X}^T\mathbf{X})^{-1}a}} \\ &= a^T\hat{\beta} \pm \sigma\sqrt{\chi^2_4(0.95)a^T(\mathbf{X}^T\mathbf{X})^{-1}a} \end{aligned}$$

Thus, we see that the bands of this method are

$$|CI^a_{ii}|/|CI^a_i| = \sqrt{\chi^2_4(0.95)}/z_{.975} = 1.57$$

times wider than those of the pointwise intervals. This is the price paid for simultaneity. Figure 3 shows an example of the two confidence bands.

We have presented here an analytical approach to this problem. One could also have "simulated" the Method 2 confidence bands by—for example—generating a large number of β-vectors on the boundary of C_β and then for each a, finding the maximum/minimum values of $a^T\beta$ observed.

R code for Problem 5:

```
set.seed(123)
n=75
x = sort(rnorm(n))
beta = rnorm(4)
X = cbind(1,poly(x,3,raw=T))
y=X%*%beta+4*rnorm(n)

bhat = solve(t(X)%*%X,t(X))%*%y
yhat = X%*%bhat
sighat = sqrt(sum((y-yhat)^2)/(n-4))
cov.bhat = solve(t(X)%*%X)*sighat^2
```

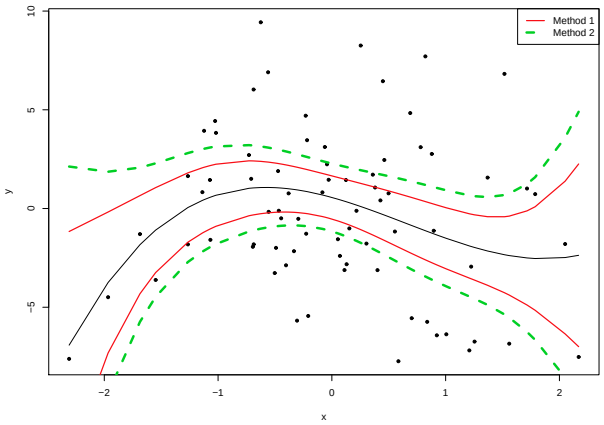

Figure 3: Problem 5

```
12  z = qnorm(1-0.025)
13
14  # Method 1
15  width = z *  sqrt(diag(X%*%cov.bhat%*%t(X)))
16  lower = yhat - width
17  upper = yhat + width
18
19  postscript("hw1_fig_p5.eps")
20  plot(x,y,pch=20)
21  lines(x,yhat)
22  lines(x,lower,lwd=2,col=2)
23  lines(x,upper,lwd=2,col=2)
24
25  # Method 2
26  qchi2 = qchisq(.95,df=4)
27  width = sqrt(qchi2 * diag(X%*%cov.bhat%*%t(X)))
28  lower = yhat - width
29  upper = yhat + width
30
31  lines(x,lower,lty=2,lwd=4,col=3)
32  lines(x,upper,lty=2,lwd=4,col=3)
33  legend("topright",legend = c("Method 1","Method 2"),col=2:3,
34         lty=1:2,lwd=c(2,4))
35  dev.off()
```

Problem 6

(a)

Since $p \gg N$, *except for those cases where the system* $(X^TX)\hat{\beta} = X^TY$ *is inconsistent*, the system will have more variables than the number of equations, which implies infinit number of solutions. The solutions form a linear space. For any $\hat{\beta}$, the residuals will be all zero.

(b)

X^TX is clearly positive semi-definite. Therefore as long as $\lambda > 0$, the matrix $X^TX + \lambda I$ is positive definite, which implies that it is invertible. Hence the ridge regression always have unique solution for $\lambda > 0$.

(c)

In ridge regression we are solving the problem

$$\min ||y - X\beta||^2 \text{ subject to } ||\beta|| < s,$$

with s inversely proportional to λ. As we let s grow, we restrict the solution less, until eventually we touch the subspace where the residuals are all 0. Because of the way in which we let the solutions grow, this will be the zero-residual solution with minimum $||\beta||$. The proof follows from the derivation of $\lim_{\lambda \downarrow 0} \hat{\beta}_\lambda^{\text{ridge}}$ in part (d).

(d)

Let the full(fat) SVD of X be UDV^T. Then the ridge regression solution is

$$\begin{aligned}
\hat{\beta}_\lambda^{ridge} =& (X^TX + \lambda I)^{-1}X^TY && (3)\\
=& (VD^TDV^T + \lambda I)^{-1}VD^TU^TY && (4)\\
=& (V(D^TD + \lambda I)^{-1}V^T)VD^TU^TY && (5)\\
=& V(D^TD + \lambda I)^{-1}D^TU^TY && (6)
\end{aligned}$$

Note that $(D^TD + \lambda I)^{-1}D^T$ is a p by N diagonal matrix where its i^{th} diagnonal entry is

$$\frac{D_i}{D_i^2 + \lambda} \xrightarrow{\lambda \to 0} \begin{cases} \frac{1}{D_i} & D_i > 0 \\ 0 & D_i = 0 \end{cases} \tag{7}$$

Note that in (7) the limit cannot be calculated by just input $\lambda = 0$ *if some* $D_i = 0$*. But the limit still exists.*
The limit can also be seen using the generalized inverse of D, denoted as D^- Therefore we have

$$\hat{\beta}_\lambda^{\text{ridge}} \xrightarrow{\lambda \to 0} VD^-U^TY \tag{8}$$

The solution could also be derived using the reduced SVD of X. If the reduced SVD of X is $X = UDV^T$ where now U and V is no longer orthogonal matrix but D is now N by N diagonal matrix. We have

$$\hat{\beta}_{\lambda}^{\text{ridge}} \xrightarrow{\lambda \to 0} VD^{-1}U^TY \tag{9}$$

Points are deducted from answers having expressions of $X\hat{\beta}_{\lambda}^{ridge}$ instead of $\hat{\beta}_{\lambda}^{ridge}$. Also, there have been some penalization for missing the interpretation in part (c).