High-dimensional data analysis HW5

1. Solve Exercise 2.9 by simulation

2. Show optimism bias under $p = 1, 2, 5, 10$ and 20

Consider a linear regression model with p parameters, fit by least squares to a set of training data $(x_1, y_1), \dots, (x_N, y_N)$ drawn at random from a population. Let $\hat{\beta}$ be the least squares estimate. Suppose we have some test data $(\tilde{x}_1, \tilde{y}_1), \dots, (\tilde{x}_M, \tilde{y}_M)$ drawn at random from the same population as the training data.

If $R_{tr}(\beta) = \frac{1}{N}\sum_1^N (y_i - \beta^T x_i)^2$ and $R_{te}(\beta) = \frac{1}{M}\sum_1^M (\tilde{y}_i - \beta^T \tilde{x}_i)^2$, prove that

$$E\left[R_{tr}(\hat{\beta})\right] \le E\left[R_{te}(\hat{\beta})\right],$$

where the expectations are over all that is random in each expression. [This exercise was brought to our attention by Ryan Tibshirani, from a homework assignment given by Andrew Ng.]

<u>Simulation design</u>

1. In Exercise 2.9

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i, \quad i = 1, \dots, n \quad \varepsilon_i \sim N(0, \sigma^2)$$

$$Y_i^0 = \beta_0 + \beta_1 X_i^0 + \varepsilon_i, \quad i = 1, \dots, n \quad \varepsilon_i^0 \sim N(0, \sigma^2)$$

Setting $\mathrm{N} = 50$, $\mathrm{M} = 30$, $\sigma^2 = 2$, $\beta = 0.5I$,

$$X_{ii} = 1, \qquad X_{i+c,i} = X_{i,i+c} = 1 + 0.1c, \qquad i = 1, \dots, N, c = 1, \dots, N$$

$$X_{ii}^0 = 1, \qquad X_{i+c,i}^0 = X_{i,i+c}^0 = 1 + 0.05c, \qquad i = 1, \dots, M, c = 1, \dots, M$$

Ex: $N = 4$

$$X = \begin{bmatrix} 1 & 1.1 \\ 1.1 & 1 \\ 1.2 & 1.1 \\ 1.3 & 1.2 \end{bmatrix}$$

We choose $p = 1$ and with 100 training set to simulate

$$E\left[R_{tr}(\hat{\beta})\right] = E\left[\frac{1}{N}\sum\nolimits_1^N \left(y_i - \hat{\beta}^T x_i\right)^2\right] = 3.944697$$

$$E\left[R_{te}(\hat{\beta})\right] = E\left[\frac{1}{M}\sum\nolimits_1^M \left(\tilde{y}_i - \hat{\beta}^T \tilde{x}_i\right)^2\right] = 4.108689$$

2.

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} + \varepsilon_i, \quad i = 1, \ldots, n \quad \varepsilon_i \sim N(0, \sigma^2)$$

$$Y_i^0 = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} + \varepsilon_i, \quad i = 1, \ldots, n \quad \varepsilon_i \sim N(0, \sigma^2)$$

Setting $\mathrm{N} = 50,\ \sigma^2 = 2,\ \beta = 0.5I,\ p = 1, \ldots, 20$

$$X_{ii} = 1, \qquad X_{i+c,i} = X_{i,i+c} = 1 + 0.1c, \qquad i = 1, \ldots, N, c = 1, \ldots, N$$

In this design,

$$op = Err_{in} - \overline{err}, \qquad where\ \ Err_{in} = E\left[R_{tr}(\hat{\beta})\right] \quad \overline{err} = E\left[R_{te}(\hat{\beta})\right]$$

Ex: $N = 4\ \ p = 4$

$$X = \begin{bmatrix} 1 & 1.1 & 1.2 & 1.3 \\ 1.1 & 1 & 1.1 & 1.2 \\ 1.2 & 1.1 & 1 & 1.1 \\ 1.3 & 1.2 & 1.1 & 1 \end{bmatrix}$$

For $p = 1, 2, 5, 10$ and 20 and with 1000 training set to simulate,

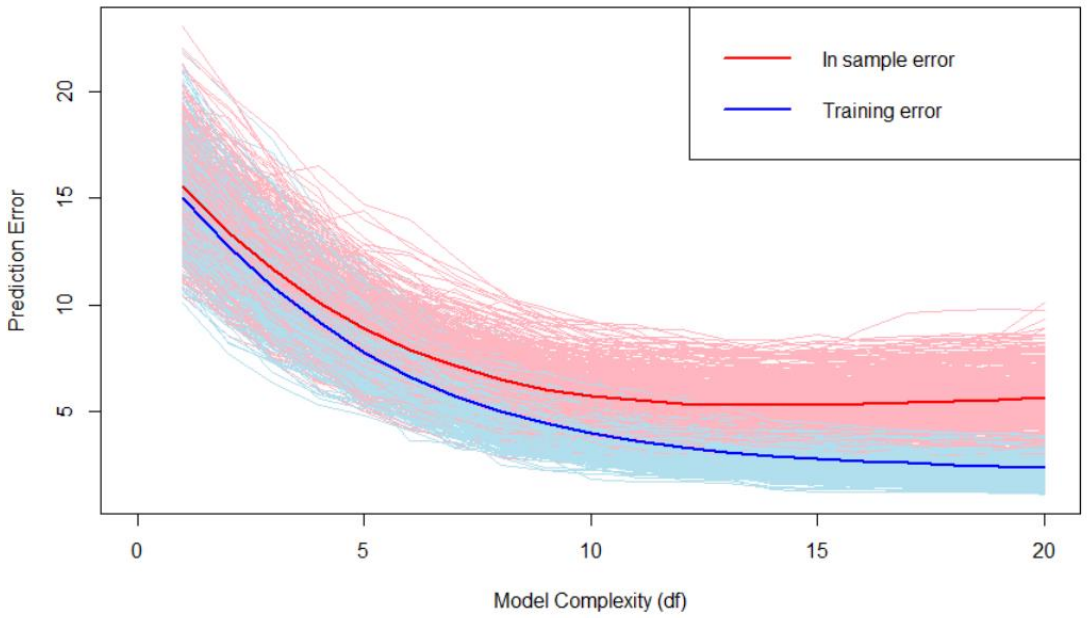
$E\left[R_{tr}(\hat{\beta})\right]_{p=1} = 15.04478 \quad E\left[R_{te}(\hat{\beta})\right]_{p=1} = 15.547877 \quad E(op) = 0.5030938$

$E\left[R_{tr}(\hat{\beta})\right]_{p=2} = 12.751866 \quad E\left[R_{te}(\hat{\beta})\right]_{p=2} = 13.421110 \quad E(op) = 0.6692438$

$E\left[R_{tr}(\hat{\beta})\right]_{p=5} = 7.775754 \quad E\left[R_{te}(\hat{\beta})\right]_{p=5} = 8.862511 \quad E(op) = 1.0867575$

$E\left[R_{tr}(\hat{\beta})\right]_{p=10} = 3.936645 \quad E\left[R_{te}(\hat{\beta})\right]_{p=10} = 5.733719 \quad E(op) = 1.7970741$

$E\left[R_{tr}(\hat{\beta})\right]_{p=20} = 2.332289 \quad E\left[R_{te}(\hat{\beta})\right]_{p=20} = 5.618822 \quad E(op) = 3.2865330$



The figure behavior of in sample error and training sample error as the model complexity is varied. The light blue curves show the training error $\overline{err}$, while the light red curves show the conditional test error Err_{in} for 1000 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected In sample error $E[Err_{in}]$ and the expected training error $E[\overline{err}]$.

Output

```
> ts-tr
 [1] 0.5030938 0.6692438 0.8371331 0.9174289 1.0867575 1.2788593 1.4032568 1.5253983 1.5988041 1.7970741
[11] 1.9082125 2.0681253 2.2140928 2.3582534 2.5253158 2.6726612 2.8446583 2.9927867 3.1386594 3.2865330
> tr
 [1] 15.044783 12.751866 10.795364  9.169782  7.775754  6.611615  5.731802  4.987359  4.426681  3.936645
[11]  3.598665  3.307390  3.080745  2.913832  2.775433  2.663818  2.565976  2.484078  2.405597  2.332289
> ts
 [1] 15.547877 13.421110 11.632497 10.087211  8.862511  7.890474  7.135059  6.512758  6.025485  5.733719
[11]  5.506877  5.375515  5.294838  5.272085  5.300749  5.336479  5.410635  5.476865  5.544256  5.618822
```

R-Code

```
### Setting
N=50
Y=rep(NA,N)


### Simulation1
nMC=100
M=30
p=1
ts_err=rep(NA,nMC)
tr_err=rep(NA,nMC)
i=1
for(i in 1:nMC){
   ### create data
   X=matrix(NA,N,p)
   for(r in 1:ncol(X)){
      for(q in 1:nrow(X)){


         for(pp in 0:50){
            if(q==(r+pp)) X[q,r]=1+0.1*pp
            if(r==(q+pp)) X[q,r]=1+0.1*pp
         }


      }
   }
   X=cbind(1,X)
   #X0=X[1:20,]
```

```
  X0=matrix(NA,M,p)
  for(r in 1:ncol(X0)){
    for(q in 1:nrow(X0)){

      for(pp in 0:50){
        if(q==(r+pp)) X0[q,r]=1+0.2*pp
        if(r==(q+pp)) X0[q,r]=1+0.2*pp
      }
    }
  }
  X0=cbind(1,X0)
  beta=rep(0.5,p+1)
  Y=X%*%beta+rnorm(N,mean=0,sd=2)
  Y0=X0%*%beta+rnorm(M,mean=0,sd=2)
  ### LSE

    beta_hat=solve(t(X)%*%X)%*%t(X)%*%Y
    tr_err[i]=mean((Y-X%*%beta_hat)^2)
    ts_err[i]=mean((Y0-X0%*%beta_hat)^2)
  }
tr=mean(tr_err)
ts=mean(ts_err)

### Simulation2
nMC=1000
p=20
M=50
```

```
ts_err=matrix(NA,nMC,p)
tr_err=matrix(NA,nMC,p)

for(i in 1:nMC){
  ### create data
  X=matrix(NA,N,p)
  for(r in 1:ncol(X)){
    for(q in 1:nrow(X)){

      for(pp in 0:50){
        if(q==(r+pp)) X[q,r]=1+0.1*pp
        if(r==(q+pp)) X[q,r]=1+0.1*pp
      }

    }
  }
  X=cbind(1,X)
  #X0=X[1:20,]
  X0=matrix(NA,M,p)
  for(r in 1:ncol(X0)){
    for(q in 1:nrow(X0)){

      for(pp in 0:50){
        if(q==(r+pp)) X0[q,r]=1+0.1*pp
        if(r==(q+pp)) X0[q,r]=1+0.1*pp
      }
```

```
    }
  }
  X0=cbind(1,X0)
  beta=rep(0.5,p+1)
  Y=X%*%beta+rnorm(N,mean=0,sd=2)
  Y0=X0%*%beta+rnorm(M,mean=0,sd=2)
  ### LSE in each p
  for(j in 1:p){
    beta_hat=solve(t(X[,(1:(j+1))])%*%X[,(1:(j+1))])%*%t(X[,(1:(j+1))])%*%Y
    tr_err[i,j]=mean((Y-X[,(1:(j+1))]%*%beta_hat)^2)
    ts_err[i,j]=mean((Y0-X0[,(1:(j+1))]%*%beta_hat)^2)
  }
}
tr=colMeans(tr_err)
ts=colMeans(ts_err)

### plot
plot(c(0,20),c(min(tr_err),max(ts_err)),type="n",xlab="Model Complexity
(df)",ylab="Prediction Error")
for(a in 1:nrow(tr_err)){
  points(1:20,tr_err[a,],type="l",col="lightblue2")
  points(1:20,ts_err[a,],type="l",col="lightpink")
}
points(1:20,tr,type="l",col="blue",lwd="2")
points(1:20,ts,type="l",col='red',lwd="2")
legend("topright",c("In sample error","Training
error"),lty=1,lwd=2,col=c("red","blue"))
```