Arif Ali
Analytics 512 Statistical Learning
May 12, 2016

Final Exam take home portion

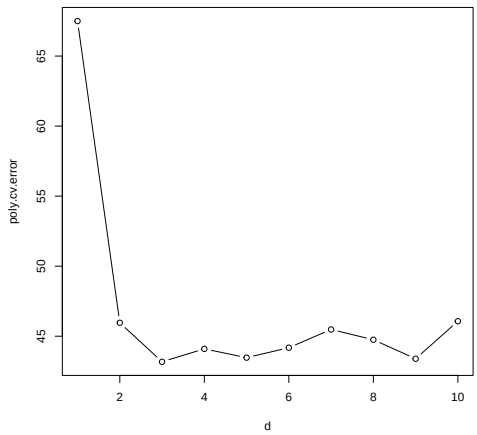
```
library("mlbench")
data(Ozone)
Ozone = as.data.frame(mapply(as.numeric, Ozone))
```

Exercise 1

```
set.seed(1933)
names(Ozone) <- c("mo","day","wday","maxoz","pressh","wind","hum",
                  "temp1","temp2","inverh","pressg","invert","vis")
Ozone$time = 1:366
Ozone = na.omit(Ozone)
train = sample(nrow(Ozone), nrow(Ozone)*.70)
Ozone_train = Ozone[train,]
```

Exercise 2

```
library(boot)
poly.cv.error = c()
d = 1:10
for(i in d){
  ozone_pm = glm(maxoz~poly(time,i), data = Ozone_train)
  poly.cv.error[i] = cv.glm(Ozone_train, ozone_pm, K = 10)$delta[2]
}
plot(d,poly.cv.error,type="b")
```

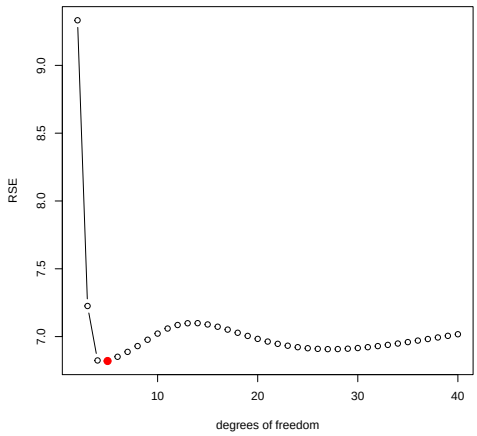


```
ozone_pm = lm(maxoz~poly(time,d[poly.cv.error == min(poly.cv.error)]), data = Ozone_train)
RSS = sum((predict(ozone_pm, Ozone[-train,])-Ozone[-train,"maxoz"])^2)
RSE = sqrt(RSS/(nrow(Ozone[-train,])-(ncol(Ozone_train)-1)-1))
```

Cross Validation identitifed 3 as the degree of polynomial best suited for predicting Max Ozone. The test Residual Standard Error is 7.0288438.

Exercise 3

```
library(splines)
RSE = 2:40
for(i in 2:40){
  ozone_smoothing.splines = smooth.spline(Ozone_train$time,
                                          Ozone_train$maxoz, df = i)
   RSS = sum((predict(ozone_smoothing.splines,
                      Ozone[-train,"time"])$y-Ozone[-train,"maxoz"])^2)
   RSE[i-1] = sqrt(RSS/(nrow(Ozone[-train,])-(ncol(Ozone_train)-1)-1))
}
plot(2:40, RSE, xlab = "degrees of freedom", type = "b")


points((which.min(RSE)+1),
       RSE[which.min(RSE)],
       col="red",cex=2,pch=20)
```

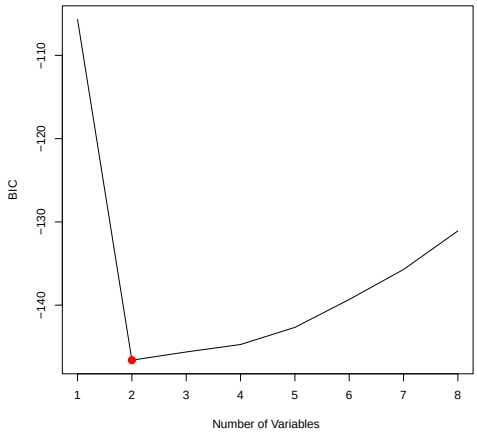


Using the Validation set approach, a smooth spline with 5 degrees of freedom is best suited for predicting Max Ozone. The test Residual Standard Error at 5 degrees of freedom is 6.8199769. It's interesting to see such a sudden drop in RSE frome 2 to 5 degrees of freedom, but after-wards, a wave like structure occur. I wondering if more than 40 degrees of freedom were explored, if the wave would just continue.

Exercise 4

```
library(leaps)
ozone_best_subset = regsubsets(maxoz~.,Ozone_train[,c("maxoz","pressh","wind","hum",
                  "temp1","temp2","inverh","pressg","invert","vis")])
```

```
plot(summary(ozone_best_subset)$bic ,
     xlab="Number of Variables ",ylab="BIC", type='l')
points(which.min(summary(ozone_best_subset)$bic),
       summary(ozone_best_subset)$bic[which.min(summary(ozone_best_subset)$bic)],
       col="red",cex=2,pch=20)
```

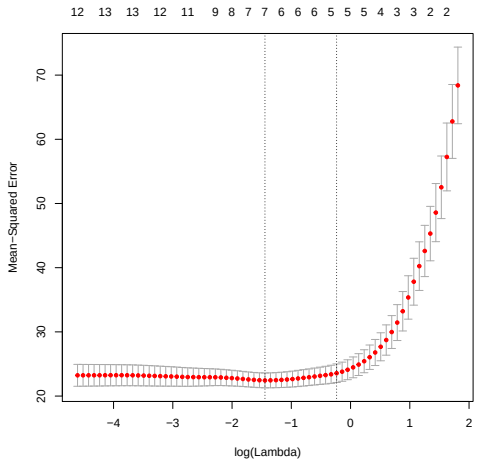



```
model = coef(ozone_best_subset ,which.min(summary(ozone_best_subset)$bic))[-1]


best_oz_model = glm(maxoz~., Ozone_train[, c(4, which(names(Ozone) %in% names(model)))], family = "gauss

RSS = sum((predict(best_oz_model, Ozone[-train, which(names(Ozone) %in% names(model))])-Ozone[-train,4]
RSE = sqrt(RSS/(nrow(Ozone[-train,])-(ncol(Ozone_train)-1)-1))
```

Based in the plot, the subset with the losest BIC is at 2 Best subset selection identified a model with the coefficients: hum,temp2 as the best performing model based on BIC. The model was then used to create a glm object, with equivilant coefficients: 0.141338354493428,0.472833174677559 with a test RSE of 5.1263292

Exercise 5

```
library(glmnet)

## Loading required package:  Matrix
## Loading required package:  foreach
## Loaded glmnet 2.0-2

grid = 10^seq(10,-2,length=100)
x = model.matrix(maxoz~.,Ozone)[,-1]
y = Ozone$maxoz
Ozone_lasso = glmnet(x[train ,], y[train], alpha = 1, lambda = grid)
cv.out=cv.glmnet(x[train ,],y[train],alpha=1)
plot(cv.out)
```




```
bestlam=cv.out$lambda.min


RSS = sum((predict(Ozone_lasso,s=bestlam,
                                newx=x[-train,])-Ozone[-train, "maxoz"])^2)
RSE = sqrt(RSS/(nrow(Ozone[-train,])-(ncol(Ozone_train)-1)-1))

out=glmnet(x[train ,],y[train],alpha=1,lambda=grid)
lasso.coef=predict(out,type="coefficients",s=bestlam)
lasso.coef[lasso.coef[,1]!=0,]

##   (Intercept)            mo           hum         temp1         temp2
## -1.635775e+01 -1.525592e-01  1.052727e-01  1.040610e-01  3.177951e-01
##        inverh        pressg           vis
## -3.234902e-04  9.555879e-04 -6.466298e-03
```

The best parameter identitified was $\lambda = 0.235829$

Exercise 6

```
library(tree)
library(gbm)

## Loading required package:  survival
##
## Attaching package:  'survival'
##
## The following object is masked from 'package:boot':
##
##    aml
##
## Loading required package:  lattice
##
## Attaching package:  'lattice'
##
## The following object is masked from 'package:boot':
```

```
##
##    melanoma
##
## Loading required package:  parallel
## Loaded gbm 2.1.1
```

```
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

Bonus

Exercise 7

```
hepatitis <- read.csv("hepatitis.data", header=FALSE, na.strings = "?")

training = sample(nrow(hepatitis), 105)

hep_train = hepatitis[training,]


hep_scale = scale(hep_train)
hc.out=hclust(dist(hep_scale),  method="average")
hc.clusters=cutree(hc.out,2)
table(hc.clusters,hep_train$V1)
```

```
##
## hc.clusters  1  2
##           1 21 83
##           2  0  1
```

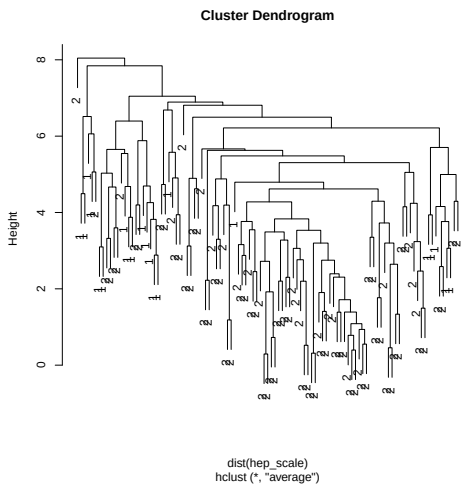
```
plot(hc.out, labels=hepatitis[training,1])

hc.out=hclust(dist(hep_scale),  method="complete")
hc.clusters=cutree(hc.out,2)
table(hc.clusters,hep_train$V1)
```

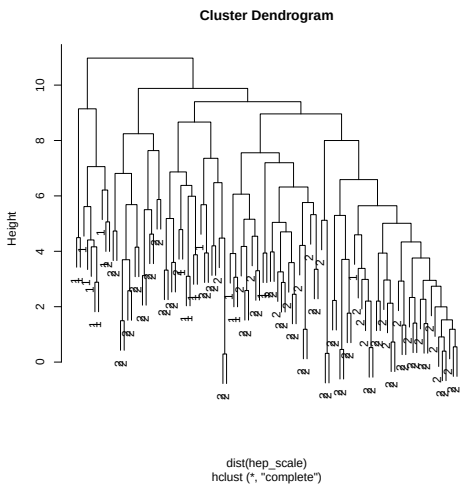
```
##
## hc.clusters  1  2
##           1 12 83
##           2  9  1
```

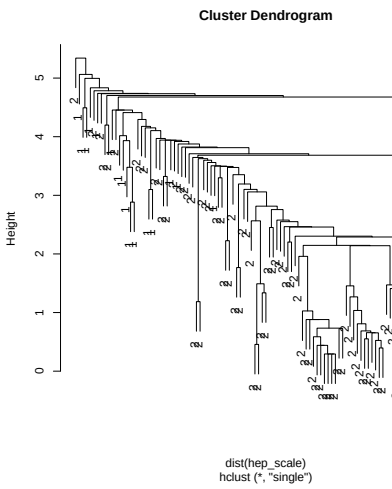
```
plot(hc.out, labels=hepatitis[training,1])

hc.out=hclust(dist(hep_scale),  method="single")
hc.clusters=cutree(hc.out,2)
table(hc.clusters,hep_train$V1)
```

```
##
## hc.clusters  1  2
##           1 21 83
##           2  0  1
```

```
plot(hc.out, labels=hepatitis[training,1])
```

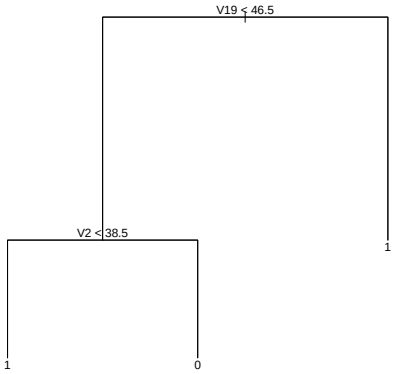






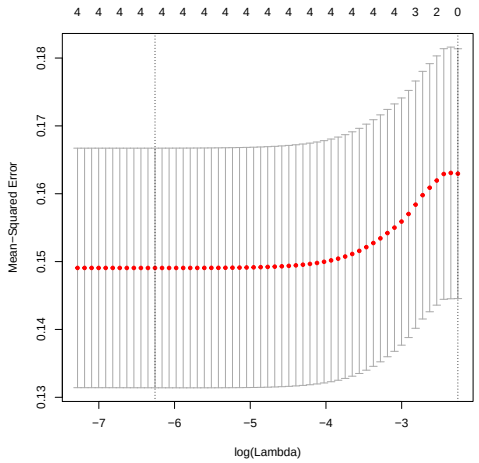

```
hepatitis$V1 = as.factor(hepatitis$V1-1)
hep_train = hepatitis[training,]


hep_boost = gbm(V1~.,data = hep_train, distribution = "multinomial", n.trees=1000)
yhat.boost=as.data.frame(predict(hep_boost,newdata=hepatitis[-training,], n.trees=1000,  type = "respons
boot_results = 1:50
boot_results[yhat.boost[,1]>yhat.boost[,2]]=0
boot_results[yhat.boost[,2]>yhat.boost[,1]]=1

mean(boot_results != hepatitis$V1[-training])

## [1] 0.2

table(boot_results, hepatitis$V1[-training])

##
## boot_results  0  1
##            0  1  0
##            1 10 39
```

```
hep_tree = tree(formula = V1 ~ ., data = hep_train)
summary(hep_tree)

##
## Classification tree:
## tree(formula = V1 ~ ., data = hep_train)
## Variables actually used in tree construction:
## [1] "V19" "V2"
## Number of terminal nodes:  3
## Residual mean deviance:  0.1834 = 9.535 / 52
## Misclassification error rate: 0.03636 = 2 / 55

plot(hep_tree)
text(hep_tree ,pretty =0)
```




```
tree_pred = predict(hep_tree, hepatitis[-training,], type = "class")
mean(tree_pred != hepatitis[-training,1])

## [1] 0.16

table(tree_pred, hepatitis[-training,1])

##
## tree_pred  0  1
##         0  5  2
##         1  6 37
```

```
mapply(function(x){sum(is.na(x))}, hepatitis[,-1])

##  V2  V3  V4  V5  V6  V7  V8  V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19
##   0   0   1   0   1   1   1  10  11   5   5   5   5   6  29   4  16  67
## V20
##   0

x = model.matrix(V1~V2+V3+V5+V20,hepatitis)[,-1]
y = as.numeric(hepatitis$V1)
hep_lasso = glmnet(x[training,],y[training], family = "binomial",alpha=1, lambda=grid)

cv.out=cv.glmnet(x[training ,],y[training],alpha=1)
plot(cv.out)
```




```
bestlam=cv.out$lambda.min
lasso.pred=as.numeric(predict(hep_lasso,s=bestlam ,newx=x[-training,], type = "class")[,1])

mean(lasso.pred != y[-training])

## [1] 0.22

table(lasso.pred, y[-training])

##
## lasso.pred  1  2
##          1  1  1
##          2 10 38

out=glmnet(x,y,alpha=1,lambda=grid)
lasso.coef=predict(out,type="coefficients",s=bestlam)
lasso.coef

## 5 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)  2.233137835
## V2          -0.004893103
## V3           0.145776784
## V5          -0.047656564
## V20         -0.214164730
```