STAT2450\_Assignment#8

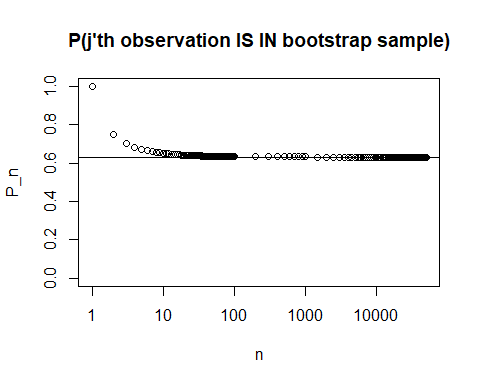
James MacPhee

B00768516

#1

#2  
(a)  
The probability that a particular observation is chosen is , so the probability it won’t be chosen is . Therefore the probability the first bootstrap observation isn’t the j’th original observation is   
(b)  
The probability is the same for the second bootstrap observation, first bootstrap observation, and everyone bootstrap observation because a bootstrap samples with replacement.  
(c)  
The probability of not choosing a particular observation would be the for everytime it is not picked, which is times. Therefore it is   
(d)  
Like I stated in the previous part the probability that a particular observation is not chosen is therefore the probability it is chosen would be , so when that equals   
(e)  
Same as the previous part but 100 instead of 5, so   
(f)  
Same as before, so (g)

n=c(1:100,100\*c(1:10),500\*(1:100))  
Pn=1-(1-1/n)^n  
plot(n,Pn,xlab="n",ylab="P\_n",main="P(j'th observation IS IN bootstrap sample)",log="x",ylim=c(0,1))  
abline(1-1/exp(1),0)

 The plot shows what parts (d), (e), and (f) were suggesting; that as n grows larger the probability that j’th observation is in the bootstrap sample converges on a value around 6.3  
(h)

store=rep(NA, 10000)  
for(i in 1:10000) {  
 store[i]=sum(sample (1:100, rep=TRUE)==4) >0  
}  
mean(store)

## [1] 0.6321

We obtained a value of around 6.3 which is the average of the probability that the j=4’th observation is in the bootstrap sample. This corroborates the previous result.

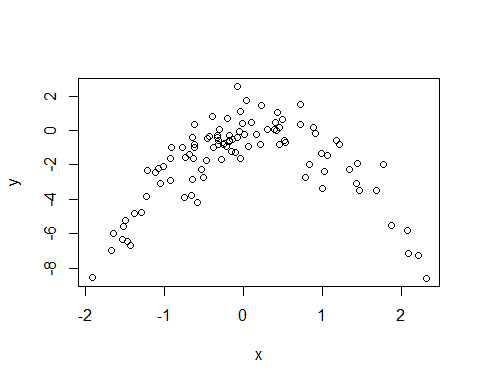
#3  
(a)  
K-fold cross validation is implemented by splitting the observations into k folds(sets) of size n/k, and then using one fold as the test set and training on the rest of the folds, then that is repeated so that each fold is used as a test set.  
(b)  
i)  
Using a simple validation set is less computationally demanding but result in more variable results depending on which observations are choosen for the test set. This approach tends to overestimate error rate. ii)  
LOOCV(leave-one-out-cross-validation) is the same as k-fold cross validation when k=n. LOOCV will result in less bias but will have greater variance and be much more computationally demanding.

#4  
(a)

set.seed(1)  
y=rnorm(100)  
x=rnorm(100)  
y=x-2\*x^2+rnorm(100)  
x2=x^2  
x3=x^3  
x4=x^4  
data=data.frame(y=y,x=x,x2=x2,x3=x3,x4=x4)

The p in question is 2, while n=100  
The model used is   
(b)

plot(x,y)

 The scatterplot shows that there is a quadratic relationship between x and y  
(c)

library(boot)  
model1=glm(y~x)  
cv.glm(data, model1)$delta[1]

## [1] 5.890979

model2=glm(y~x+x2)  
cv.glm(data, model2)$delta[1]

## [1] 1.086596

model3=glm(y~x+x2+x3)  
cv.glm(data, model3)$delta[1]

## [1] 1.102585

model4=glm(y~x+x2+x3+x4)  
cv.glm(data, model4)$delta[1]

## [1] 1.114772

set.seed(12)  
y=rnorm(100)  
x=rnorm(100)  
y=x-2\*x^2+rnorm(100)  
x2=x^2  
x3=x^3  
x4=x^4  
data=data.frame(y=y,x=x,x2=x2,x3=x3,x4=x4)  
model1=glm(y~x)  
cv.glm(data, model1)$delta[1]

## [1] 5.974045

model2=glm(y~x+x2)  
cv.glm(data, model2)$delta[1]

## [1] 1.065769

model3=glm(y~x+x2+x3)  
cv.glm(data, model3)$delta[1]

## [1] 1.093966

model4=glm(y~x+x2+x3+x4)  
cv.glm(data, model4)$delta[1]

## [1] 1.110949

1. Model (ii) had the smallest LOOCV error which matches what I expected because that is the quadratic model.

yi=lm(y~x,data=data)  
syi=coefficients(summary(yi))[2,4]  
syi

## [1] 0.0005453845

yi=lm(y~x+x2,data=data)  
syi=coefficients(summary(yi))[2:3,4]  
syi

## x x2   
## 1.512363e-14 4.187084e-38

yi=lm(y~x+x2+x3,data=data)  
syi=coefficients(summary(yi))[2:4,4]  
syi

## x x2 x3   
## 1.168308e-04 1.039054e-37 7.225558e-01

yi=lm(y~x+x2+x3+x4,data=data)  
syi=coefficients(summary(yi))[2:5,4]  
syi

## x x2 x3 x4   
## 1.674471e-04 3.426353e-08 6.431383e-01 3.997472e-01

In the linear model the coefficient of is not statistically significant.  
In the quadratic model the coefficients of and are both statistically significant.  
In the quadratic model the coefficients of and are both statistically significant while for it is not statistically significant.  
In the quadratic model the coefficients of and are both statistically significant while for and it is not statistically significant.  
These results agree with previous results because the quadratic model is significant while anything extra is not.  
#5  
Under the majority vote approach the final classification would be Red because six of the estimates have a probabiity over 0.5.  
under the average probability approach the final classification would be Green because the average proabbility is   
#6  
(a)

set.seed(77191)  
library(ISLR)  
library(randomForest)

## randomForest 4.6-14

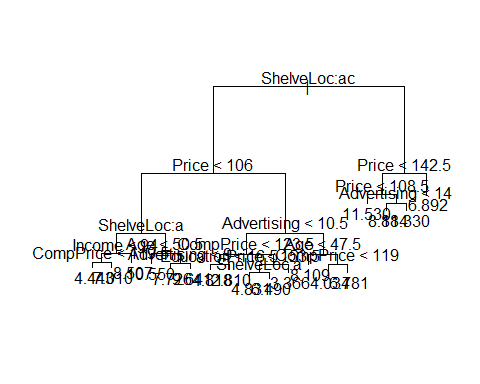
## Type rfNews() to see new features/changes/bug fixes.

attach(Carseats)  
n=nrow(Carseats)  
indices=sample(1:n,n/2,replace=F)  
cstrain=Carseats[indices,]  
cstest=Carseats[-indices,]

library(tree)  
cstree=tree(Sales~.,data=cstrain)  
summary(cstree)

##   
## Regression tree:  
## tree(formula = Sales ~ ., data = cstrain)  
## Variables actually used in tree construction:  
## [1] "ShelveLoc" "Price" "Income" "CompPrice" "Age"   
## [6] "Advertising" "Education"   
## Number of terminal nodes: 18   
## Residual mean deviance: 2.235 = 406.8 / 182   
## Distribution of residuals:  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -4.01800 -0.92800 -0.08521 0.00000 1.09400 3.71700

plot(cstree)  
text(cstree)

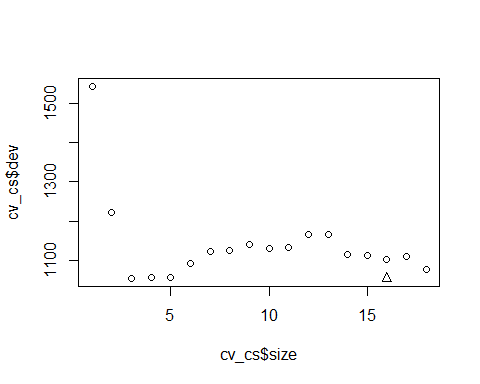


yh=predict(cstree,newdata=cstest)  
mean((yh-cstest$Sales)^2)

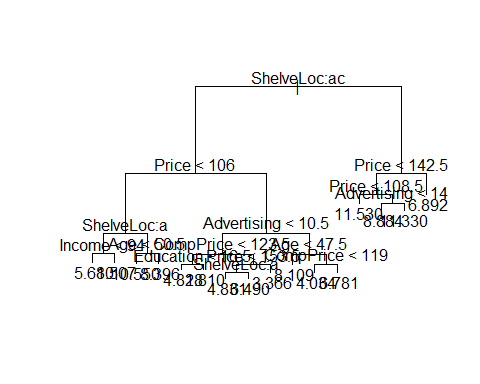
## [1] 4.274251

This tree shows that ShelveLoc and Price affect the decision to buy more than CompPrice and advertising.  
The test MSE is 4.272451  
(c)

cv\_cs=cv.tree(cstree)  
plot(cv\_cs$size,cv\_cs$dev)  
min=which.min(cv\_cs$dev)  
points(min,cv\_cs$dev[min],pch=2)



pruned=prune.tree(cstree,best=min)  
plot(pruned)  
text(pruned)



yh=predict(pruned,newdata=cstest)  
mean((yh-cstest$Sales)^2)

## [1] 4.367841

Pruning the tree actually increased MSE, so pruning did not improve it.  
(d)

bagged=randomForest(Sales~.,data=cstrain,ntree=500,importance=TRUE)  
y\_bagged=predict(bagged,newdata=cstest)  
mean((y\_bagged-cstest$Sales)^2)

## [1] 3.408087

importance(bagged)

## %IncMSE IncNodePurity  
## CompPrice 11.0914028 135.17662  
## Income 3.4633864 138.23607  
## Advertising 13.9004787 159.85733  
## Population 0.2014370 105.26464  
## Price 33.1311638 336.86417  
## ShelveLoc 37.1557340 297.71813  
## Age 9.7022898 151.55533  
## Education -1.1767904 62.14122  
## Urban 0.1718705 12.02894  
## US 4.9626950 28.73752

The MSE was improved this time, bagging is a better method than pruning in this case.  
It seems Price and ShelveLoc are the most important variables, just as I stated earlier.  
(e)

rf=randomForest(Sales~.,data=cstrain,ntree=500,importance=TRUE)  
y\_rf=predict(rf,newdata=cstest)  
mean((y\_rf-cstest$Sales)^2)

## [1] 3.372307

importance(rf)

## %IncMSE IncNodePurity  
## CompPrice 11.3732125 127.63694  
## Income 3.4620911 141.51400  
## Advertising 15.1317801 158.25057  
## Population 2.2957353 102.43108  
## Price 32.7344213 348.84675  
## ShelveLoc 36.1108240 297.73273  
## Age 9.5644010 148.72588  
## Education -0.6222804 65.45152  
## Urban -0.2547658 12.68120  
## US 5.3645021 34.15887

The MSE was improved to about the same level as bagging had acheived.  
Again the two most important variables are Price and ShelveLoc.