PREDICTION COMPETITION 1

1a .

Using a simple prediction and confidence model from chapter 2 (not validation approach)

```
## 1 a)
library(ISLR2)
library(boot)
library(B sample(x, size, replace = FALSE, prob = NULL)

train <- sample(506,253)
lm.fit <- lm(medv~rm , data = Boston, subset = train)
predict(lm.fit ,data.frame(rm = (c(5,10,15))),interval = "prediction")
predict(lm.fit ,data.frame(rm = (c(5,10,15))),interval = "confidence")

##plot(mpg~horsepower, data= Auto)
# glm.fit <- glm(mpg~horsepower, data= Auto)
# cv.err <- cv.glm(Auto, glm.fit)
# cv.err$delta
```

```
> train <- sample(506,253)
> lm.fit <- lm(medv~rm , data = Boston, subset = train)
> predict(lm.fit ,data.frame(rm = (c(5,10,15))),interval = "prediction")
       fit       lwr       upr
1 11.79818 -2.818855  26.41521
2 55.33941 40.127533  70.55129
3 98.88064 80.928630 116.83266
> predict(lm.fit ,data.frame(rm = (c(5,10,15))),interval = "confidence")
       fit       lwr       upr
1 11.79818  9.994386  13.60197
2 55.33941 50.757134  59.92169
3 98.88064 88.303742 109.45755
> 
```

Using the validation model :

```r
##1 a
library(ISLR2)
library(boot)
library(Boston)

set.seed(1)
train <- sample(506,253)
lm.fit<- lm(medv~rm, data= Boston, subset = train)
mean((medv - predict(lm.fit,Boston))[-train]^2)
## getting the error rates
lm.fit2<- lm(medv~poly(rm,2), data= Boston, subset = train)
mean((medv - predict(lm.fit2,Boston))[-train]^2)

## error rates 2
lm.fit3<- lm(medv~poly(rm,3), data= Boston, subset = train)
```

18:1 (Top Level) ÷ R Script ÷

```r
> train <- sample(506,253)
> lm.fit<- lm(medv~rm, data= Boston, subset = train)
> mean((medv - predict(lm.fit,Boston))[-train]^2)
[1] 49.65573
> ## getting the error rates
> lm.fit2<- lm(medv~poly(rm,2), data= Boston, subset = train)
> mean((medv - predict(lm.fit2,Boston))[-train]^2)
[1] 50.59916
>
> ## error rates 2
> lm.fit3<- lm(medv~poly(rm,3), data= Boston, subset = train)
> mean((medv - predict(lm.fit3,Boston))[-train]^2)
[1] 48.63831
>
```

Getting the error rates^^ 1C) using LOOCV

```r
## 1C
library(ILSR2)
attach(Boston)
library(boot)
glm.fit<- glm(medv~rm, data = Boston)
cv.err<- cv.glm(Boston, glm.fit)
cv.err$delta


## finding the ith vectors
cv.error <- rep(0,10)
for (i in 1:10) {
  glm.fit <- glm(medv~poly(rm , i), data = Boston)
  cv.error[i] <- cv.glm(Boston , glm.fit)$delta [1]
}
```

```
> glm.fit<- glm(medv~rm, data = Boston)
> cv.err<- cv.glm(Boston, glm.fit)
> cv.err$delta
[1] 44.21666 44.21605
>
>
> ## finding the ith vectors
> cv.error <- rep(0,10)
> for (i in 1:10) {
+   glm.fit <- glm(medv~poly(rm , i), data = Boston)
+   cv.error[i] <- cv.glm(Boston , glm.fit)$delta [1]
+ }
> cv.error
 [1] 44.21666 39.13461 38.24056 38.49876 36.14964 37.04616 37.66696 37.98703 67.29942 36.62635
>
```

1D)

Using K-fold LOOCV

```r
# 1D
library(ISLR2)
attach(Boston)
library(boot)
set.seed(10)
cv.error.K10 <- rep(0,10)
for (i in 1:10) {
  glm.fit <- glm(medv~poly(rm , i), data = Boston)
  cv.error.K10[i] <- cv.glm(Boston , glm.fit, K = 10)$delta [1]
}
```

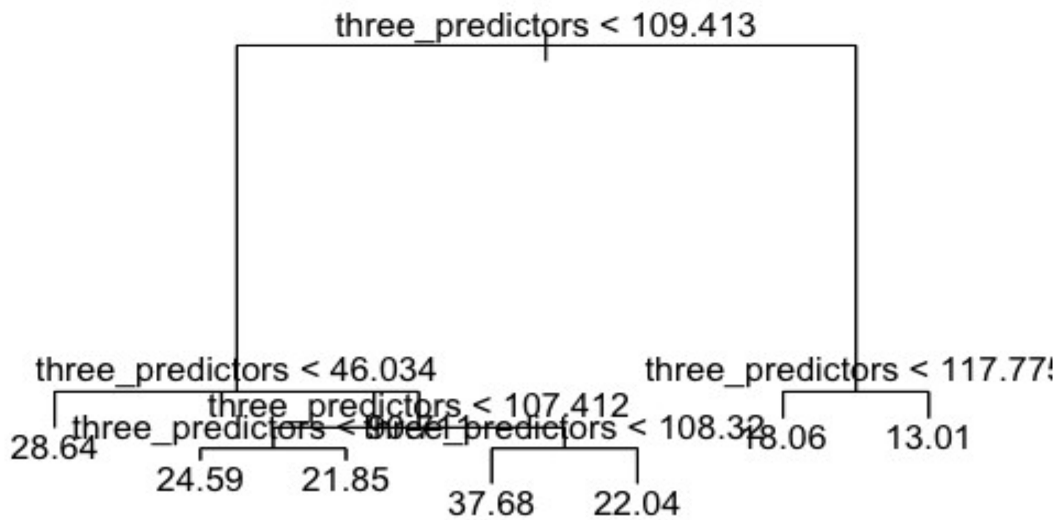
```
The following objects are masked from Boston (pos = 30):

    age, chas, crim, dis, indus, lstat, medv, nox, ptratio, rad, rm, tax, zn

> library(boot)
> set.seed(10)
> cv.error.K10 <- rep(0,10)
> for (i in 1:10) {
+   glm.fit <- glm(medv~poly(rm , i), data = Boston)
+   cv.error.K10[i] <- cv.glm(Boston , glm.fit, K = 10)$delta [1]
+ }
> cv.error.K10
 [1] 44.07360 39.15957 38.13926 38.61397 35.96672 36.28740 37.29397 38.00877 92.69203 36.83117
> 
```

2A)

```r
## 2a
library(boot)
library(tree)
library(ISLR2)
attach(Boston)


set.seed(1)
train <- sample(1:nrow(Boston), nrow(Boston)/2)
test<- - train
training_data = Boston[train,]
testing_data = Boston[test,]
testing_medv = medv[test]
three_predictors = rm + lstat +age
tree_Boston = tree(medv~three_predictors, Boston, subset = train)

cv.boston <- cv.tree(tree_Boston)
prune_boston = prune.tree(tree_Boston, best = 5)

```

```
> three_predictors = rm + lstat +age
> tree_Boston = tree(medv~three_predictors, Boston, subset = train)
>
> cv.boston <- cv.tree(tree_Boston)
> prune_boston = prune.tree(tree_Boston, best = 5)
> summary(tree_Boston)

Regression tree:
tree(formula = medv ~ three_predictors, data = Boston, subset = train)
Number of terminal nodes:  7
Residual mean deviance:  45.49 = 11190 / 246
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-18.680  -3.986  -1.209   0.000   2.462  26.950
>
```

Upon plotting, we get :



Using the cross validation :

```r
## cross validation :
yhat = predict(tree_Boston, newdata = testing_data)
boston_test = Boston[-train, "medv"]
plot(yhat,boston_test)
mean((yhat-boston_test)^2)
```

22:37 (Top Level) R Script

```
> mean((yhat-boston_test)^2)
[1] 136.5067
```

The MSE associated with the regression tree is 136.5067

2C)
Using all the variables

```r
### 2c
library(boot)
library(tree)
library(ISLR2)
attach(Boston)


set.seed(1)
train <- sample(1:nrow(Boston), nrow(Boston)/2)
test<- - train
training_data = Boston[train,]
testing_data = Boston[test,]
testing_medv = medv[test]
tree.Boston = tree(medv~., Boston, subset = train)
```

```
> testing_data = Boston[test,]
> testing_medv = medv[test]
> tree.Boston = tree(medv~., Boston, subset = train)
> summary(tree.Boston)

Regression tree:
tree(formula = medv ~ ., data = Boston, subset = train)
Variables actually used in tree construction:
[1] "rm"    "lstat" "crim"  "age"
Number of terminal nodes:  7
Residual mean deviance:  10.38 = 2555 / 246
Distribution of residuals:
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-10.1800  -1.7770  -0.1775   0.0000   1.9230  16.5800
>
```