Lab #16 - Regression and Classification Trees

Econ 224

November 1st, 2018

Introduction

In this lab you will work through Sections 8.3.1, 8.3.2, and 8.3.3 of ISL and record your code and results in an RMarkdown document. I have added section headings below to help you organize your results. You do not have to submit this lab, so you don't have to type up a detailed description of what you've done. However, I'd suggest that you write down some notes for your own future reference. These will be helpful on the problem set. You do not need to follow the code in ISL exactly: feel free to use your preferred coding style.

You will need the `ISLR`, `tree` and `randomForest` packages for this lab, so please install them if you have not done so already. This lab uses two datasets: `Carseats` which is contained in `ISLR`, and `Boston` which is contained in `MASS`.

Fitting Classification Trees

Work through section 8.3.1 of ISL and add your code and results below.

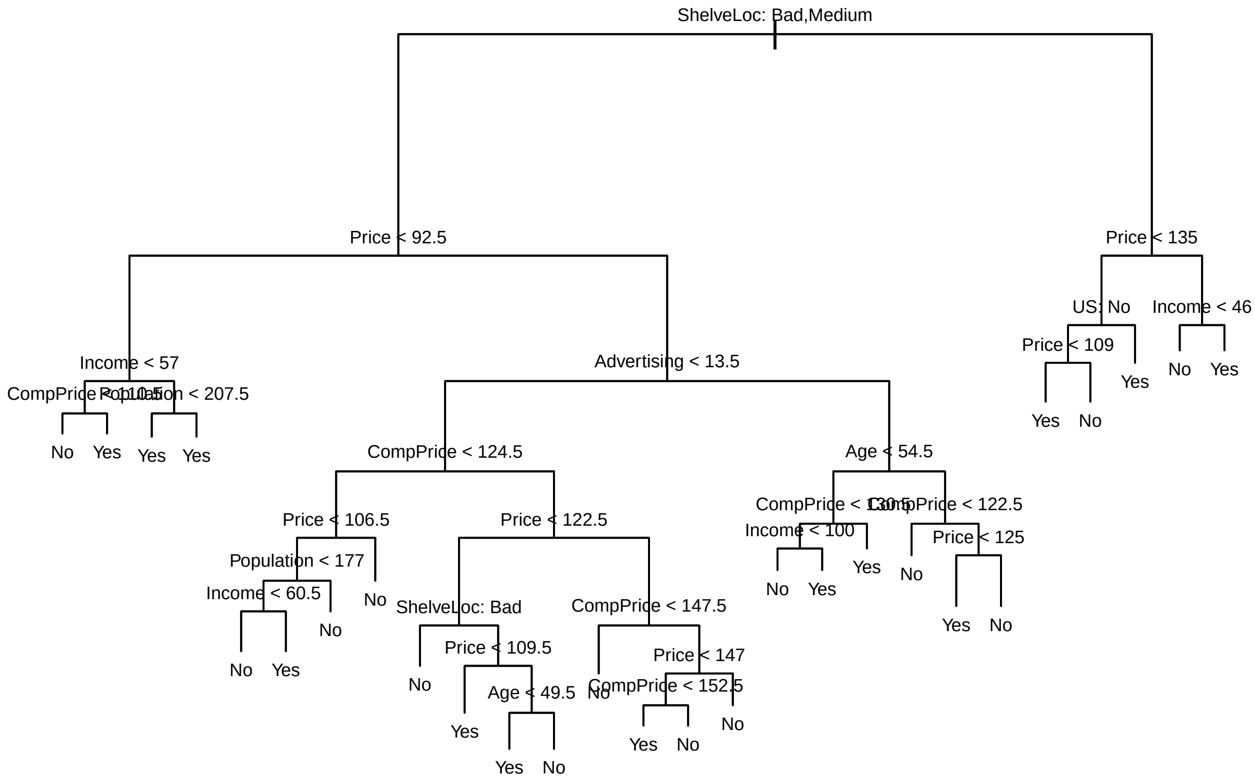
```r
library(tree)
library(ISLR)
library(dplyr)

#------------------ Create binary outcome for high or low sales
# (tree requires that High is a factor)
Carseats <- Carseats %>%
  mutate(High = as.factor(if_else(Sales <= 8, 'No', 'Yes'))) %>%
  select(-Sales)

#------------------ Fit, summarize, and plot classification tree
tree_carseats <- tree(High ~ ., Carseats)
summary(tree_carseats)
```

```
Classification tree:
tree(formula = High ~ ., data = Carseats)
Variables actually used in tree construction:
[1] "ShelveLoc"   "Price"       "Income"      "CompPrice"   "Population"
[6] "Advertising" "Age"         "US"
Number of terminal nodes:  27
Residual mean deviance:  0.4575 = 170.7 / 373
Misclassification error rate: 0.09 = 36 / 400
```

```r
plot(tree_carseats)
text(tree_carseats, pretty = 0, cex = 0.6)
```




```
#----------------- Evaluate classification performance using test dataset
set.seed(2)
train <- sample(1:nrow(Carseats), nrow(Carseats) / 2)
carseats_test <- Carseats[-train,]
high_test <- Carseats$High[-train]

tree_carseats_train <- tree(High ~ ., Carseats, subset = train)
tree_carseats_pred <- predict(tree_carseats_train, newdata = carseats_test,
                              type = 'class') # Return class predictions, not probs
table(tree_carseats_pred, high_test)
```

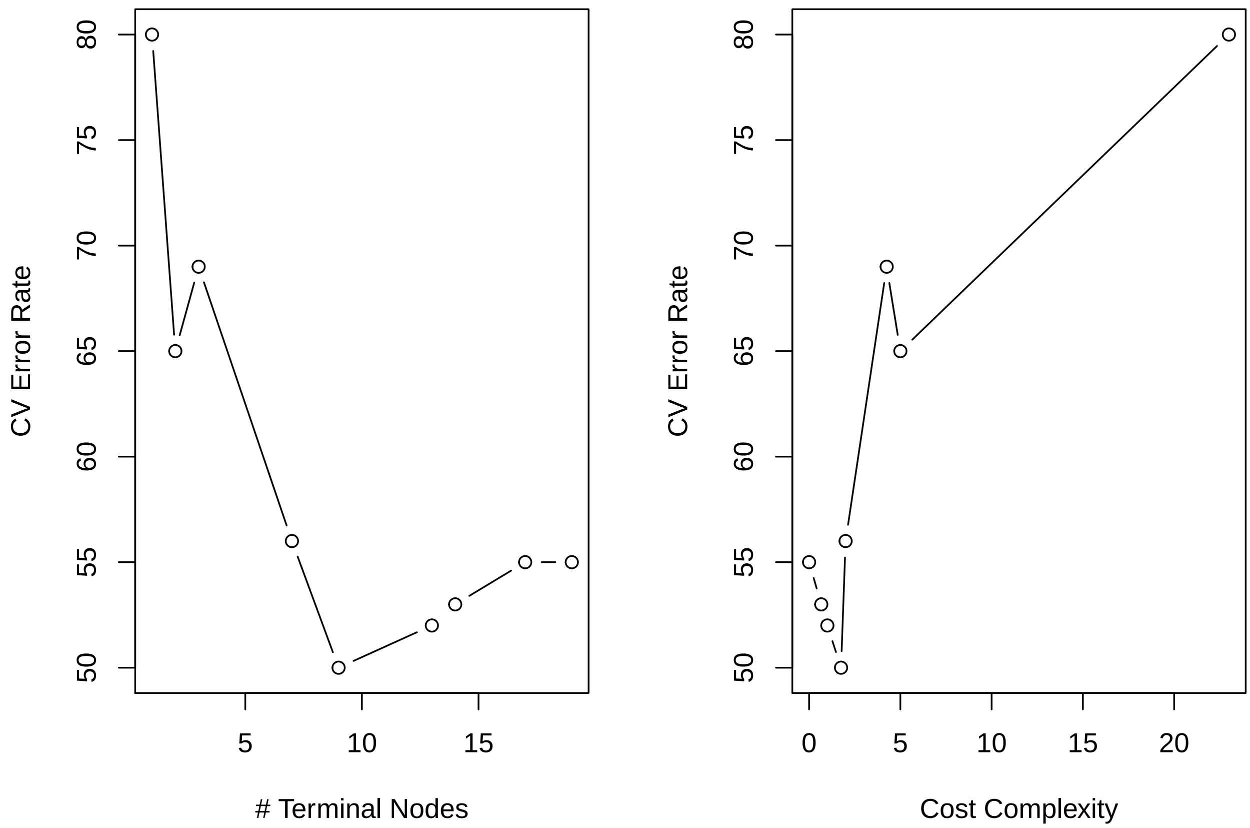
```
                  high_test
tree_carseats_pred No Yes
               No  86  27
               Yes 30  57
```

```
(86 + 57) / (86 + 57 + 30 + 27) # Correct prediction rate
```

```
[1] 0.715
```

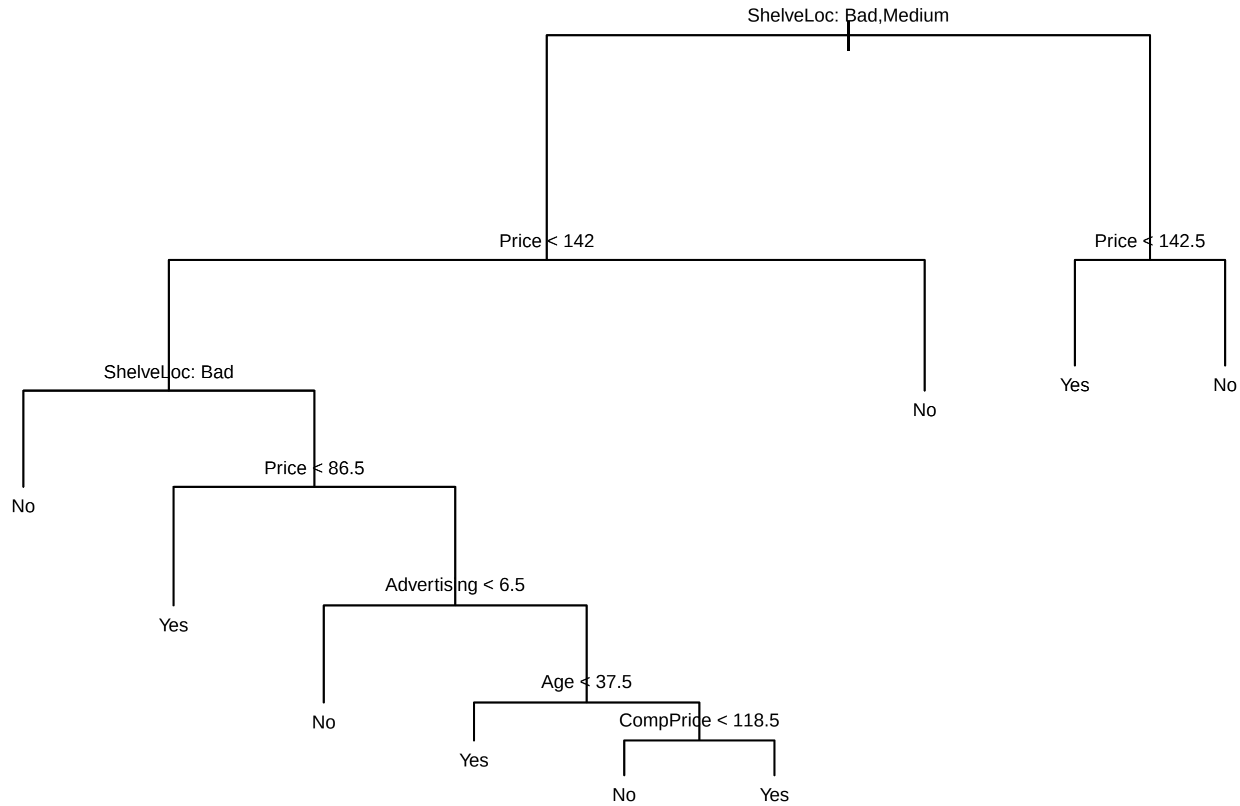
```
#----------------- Cost complexity pruning via cross-validation
# (prune using mis-classification rate rather than deviance)
set.seed(3)
cv_carseats <- cv.tree(tree_carseats_train, FUN = prune.misclass)
par(mfrow = c(1,2))
plot(cv_carseats$size, cv_carseats$dev, type = 'b', xlab = '# Terminal Nodes',
     ylab = 'CV Error Rate')
```

```r
plot(cv_carseats$k, cv_carseats$dev, type = 'b', xlab = 'Cost Complexity',
     ylab = 'CV Error Rate')
```




```r
par(mfrow = c(1, 1))

#----------------- Prune to the best tree from the CV exercise
# (from the plots, this as 9 terminal nodes)
prune_carseats <- prune.misclass(tree_carseats_train, best = 9)
plot(prune_carseats)
text(prune_carseats, pretty = 0, cex = 0.6)
```




```
prune_predict <- predict(prune_carseats, carseats_test, type = 'class')
table(prune_predict, high_test)
```

```
             high_test
prune_predict No Yes
          No  94  24
          Yes 22  60
```

```
(94 + 60) / (94 + 60 + 22 + 24)
```

```
[1] 0.77
```

```
#----------------- Clean up
rm(list = ls())
```

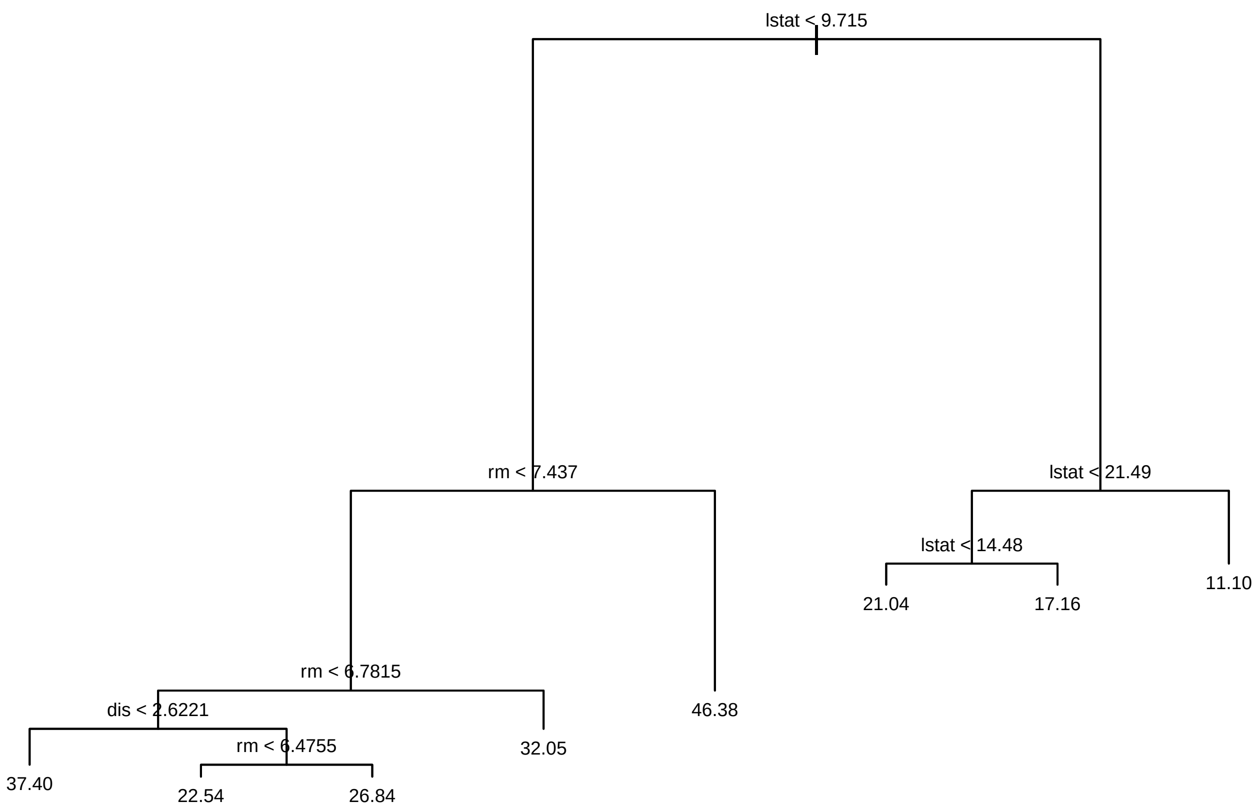
Fitting Regression Trees

Work through section 8.3.2 of ISL and add your code and results below.

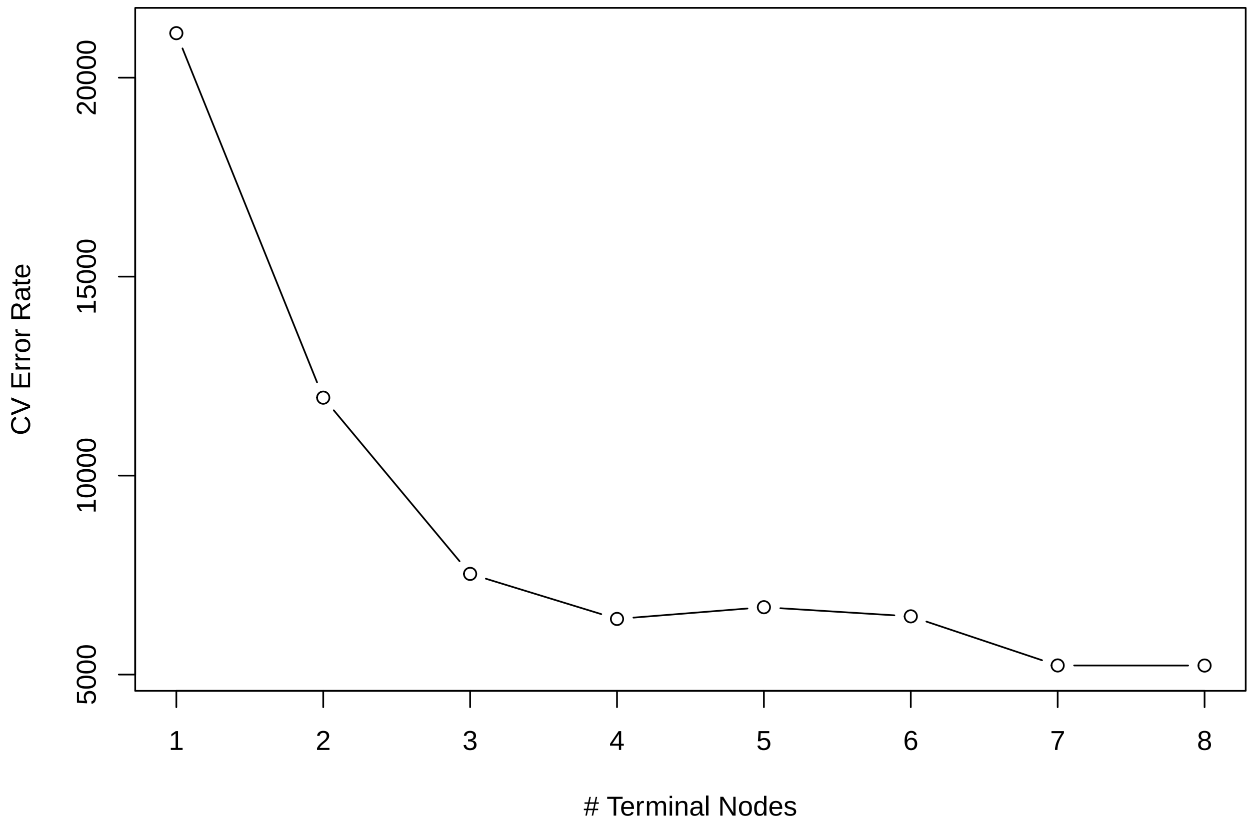
```
#----------------- Fit regression tree to training subset of Boston data
library(MASS)
set.seed(1)
boston_train <- sample(1:nrow(Boston), nrow(Boston) / 2)
tree_boston_train <- tree(medv ~., Boston, subset = boston_train)
summary(tree_boston_train)
```

```
Regression tree:
tree(formula = medv ~ ., data = Boston, subset = boston_train)
Variables actually used in tree construction:
[1] "lstat" "rm"    "dis"
Number of terminal nodes:  8
Residual mean deviance:  12.65 = 3099 / 245
Distribution of residuals:
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-14.10000  -2.04200  -0.05357   0.00000   1.96000  12.60000
```

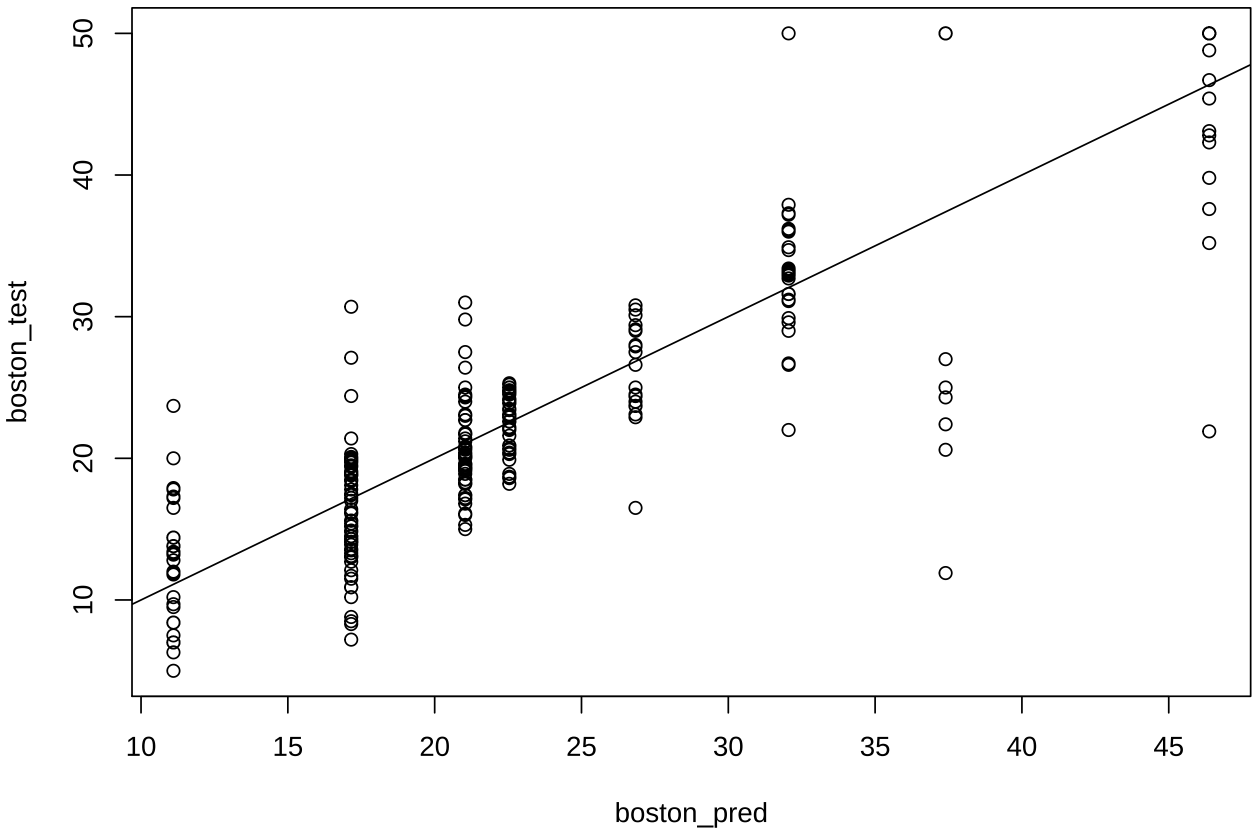
```r
plot(tree_boston_train)
text(tree_boston_train, pretty = 0, cex = 0.6)
```




```r
#----------------- Use cross-validation to prune the tree
# (But the unpruned tree comes out as the best!)
cv_boston <- cv.tree(tree_boston_train)
plot(cv_boston$size, cv_boston$dev, type = 'b', xlab = '# Terminal Nodes',
     ylab = 'CV Error Rate')
```




```r
#----------------- Test Error for Boston dataset
boston_pred <- predict(tree_boston_train, newdata = Boston[-boston_train,])
boston_test <- Boston[-boston_train, 'medv']
plot(boston_pred, boston_test)
abline(0, 1)
```




```
mean((boston_pred - boston_test)^2)
```

```
[1] 25.04559
```

Bagging and Random Forests

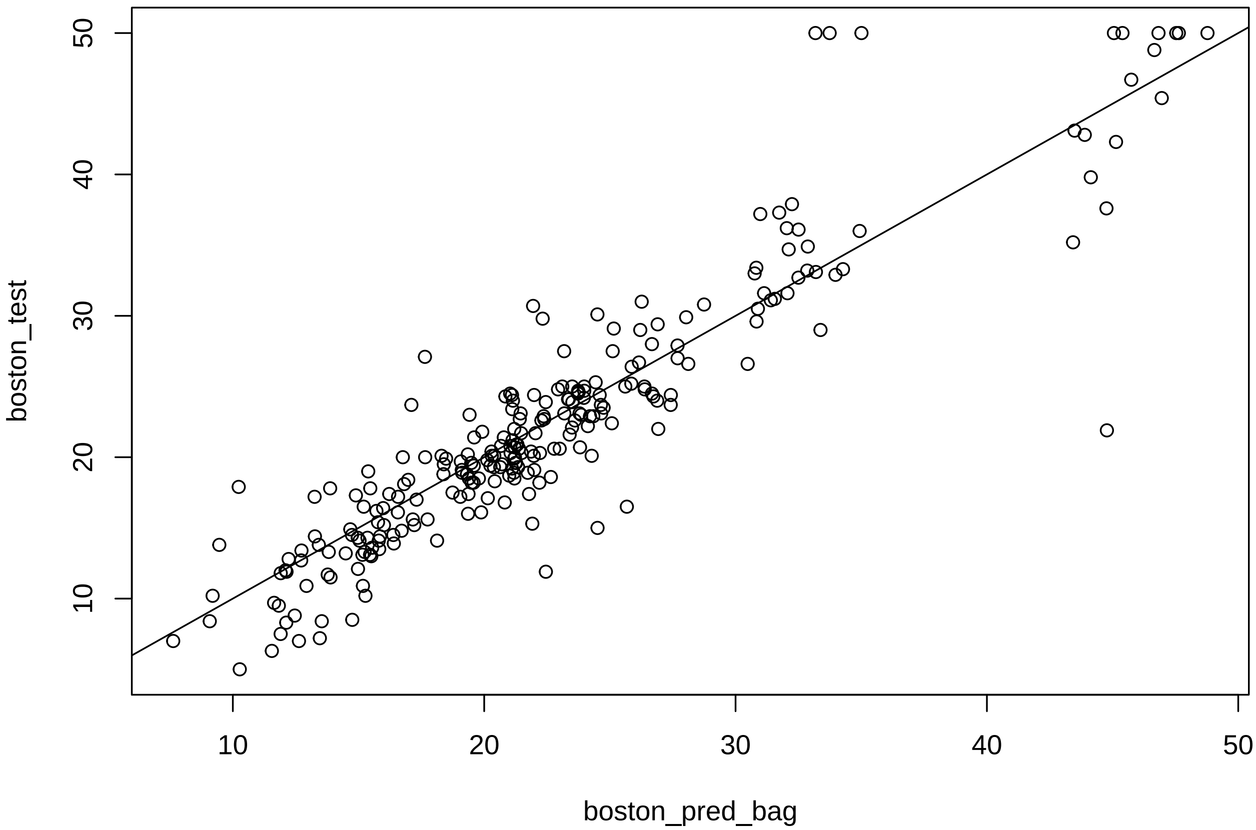
Work through section 8.3.3 of ISL and add your code and results below.

```
library(randomForest)
#------------------- Bagging for Boston dataset
# (bagging is a special case of random forests with m = p)
set.seed(1)
bag_boston <- randomForest(medv ~ ., data = Boston, subset = boston_train,
                           # can also set ntree to get a different number of trees
                           # the default is 500
                           mtry = 13, #mtry sets m (number of predictors per split)
                           importance = TRUE) # assess importance of predictors
bag_boston #slightly different results from the book
```

```
Call:
 randomForest(formula = medv ~ ., data = Boston, mtry = 13, importance = TRUE,      subset = boston_tra
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 13
```

```
          Mean of squared residuals: 11.15723
                    % Var explained: 86.49
```

```
boston_pred_bag <- predict(bag_boston, newdata = Boston[-boston_train,])
plot(boston_pred_bag, boston_test)
abline(0,1)
```




```
mean((boston_pred_bag - boston_test)^2) # slightly different result from book
```

```
[1] 13.50808
```

```
#------------------ Random Forests for Boston Dataset
# (use 6 predictors for each tree)
set.seed(1)
rf_boston <- randomForest(medv ~., data = Boston, subset = boston_train,
                          mtry = 6, importance = TRUE)
boston_pred_rf <- predict(rf_boston, newdata = Boston[-boston_train, ])
mean((boston_pred_rf - boston_test)^2)
```

```
[1] 11.66454
```

```r
importance(rf_boston)
```

```
          %IncMSE IncNodePurity
crim    12.132320     986.50338
zn       1.955579      57.96945
indus    9.069302     882.78261
chas     2.210835      45.22941
nox     11.104823    1044.33776
rm      31.784033    6359.31971
age     10.962684     516.82969
dis     15.015236    1224.11605
rad      4.118011      95.94586
tax      8.587932     502.96719
ptratio 12.503896     830.77523
black    6.702609     341.30361
lstat   30.695224    7505.73936
```

```r
varImpPlot(rf_boston)
```