

## Chapter 8

### Lab

#### Fitting Classification Trees

```
carseats <- as.data.table(ISLR::Carseats)

carseats[, High := as.factor(ifelse(Sales <= 8, "No", "Yes"))]

tree.carseats <- tree(formula = High ~ .-Sales, data = carseats)

summary(tree.carseats)
```

Classification tree:

```
tree(formula = High ~ . - Sales, data = carseats)
```

Variables actually used in tree construction:

```
[1] "ShelveLoc" "Price" "Income" "CompPrice" "Population"
```

```
[6] "Advertising" "Age" "US"
```

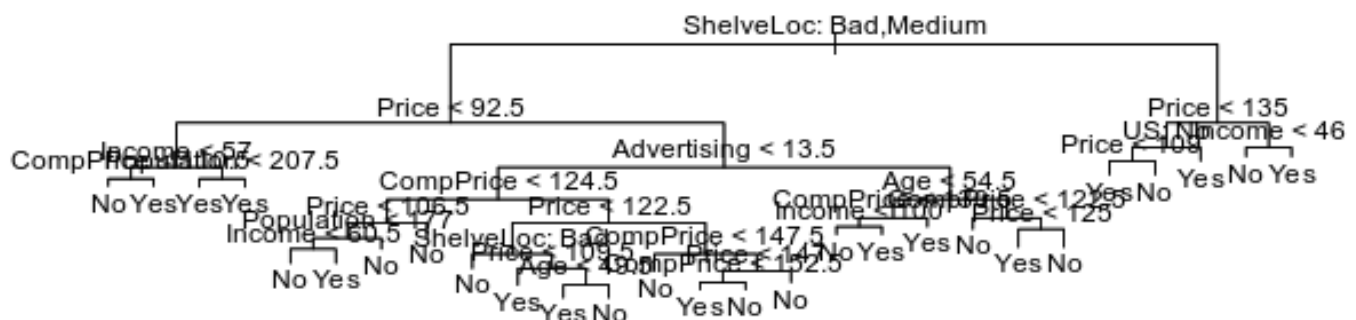
Number of terminal nodes: 27

Residual mean deviance: 0.4575 = 170.7 / 373

Misclassification error rate: 0.09 = 36 / 400

```
plot(tree.carseats)
```

```
text(tree.carseats, pretty = 0)
```



```

set.seed(2)

train <- sample(1:nrow(carseats), 200)

carseats.test <- carseats[-train]
high.test <- carseats[-train]$High

tree.carseats <- tree(High ~.-Sales, data = carseats, subset = train)
tree.pred <- predict(tree.carseats, carseats.test, type = "class")

table(tree.pred, high.test)

```

```

      high.test
tree.pred No Yes
      No  104  33
      Yes   13  50

```

```

set.seed(3)

cv.carseats <- cv.tree(tree.carseats, FUN = prune.misclass)
names(cv.carseats)

```

```

[1] "size"    "dev"      "k"        "method"

cv.carseats

$size
[1] 21 19 14  9  8  5  3  2  1

$dev
[1] 74 76 81 81 75 77 78 85 81

$k
[1] -Inf  0.0  1.0  1.4  2.0  3.0  4.0  9.0 18.0

$method
[1] "misclass"

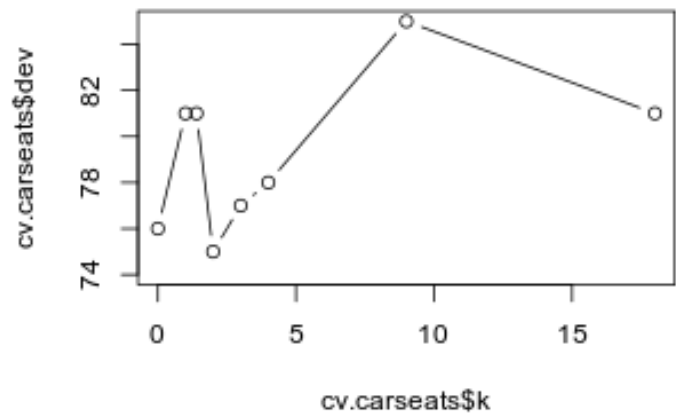
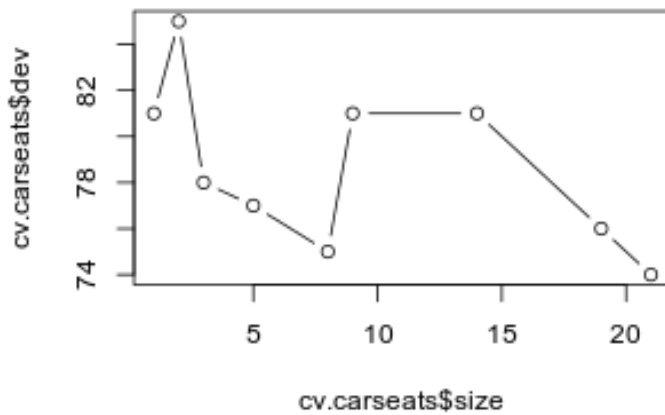
attr("class")
[1] "prune"      "tree.sequence"

```

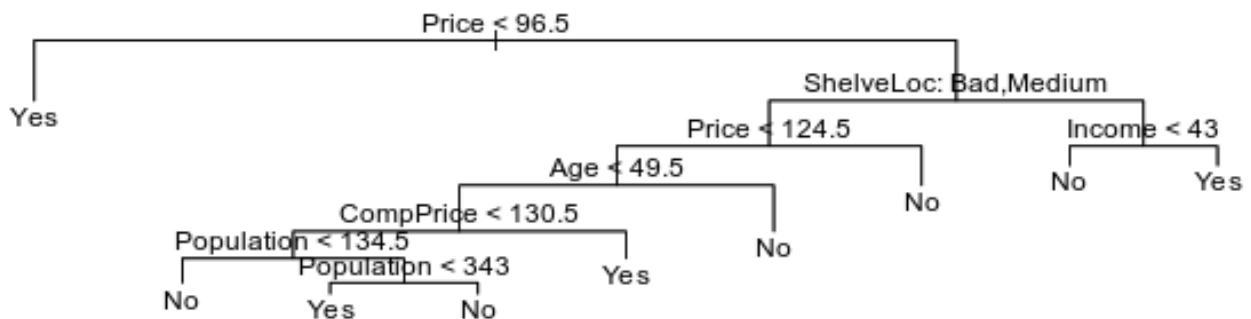
```

par(mfrow = c(1, 2))
plot(cv.carseats$size, cv.carseats$dev, type = "b")
plot(cv.carseats$k, cv.carseats$dev, type="b")

```



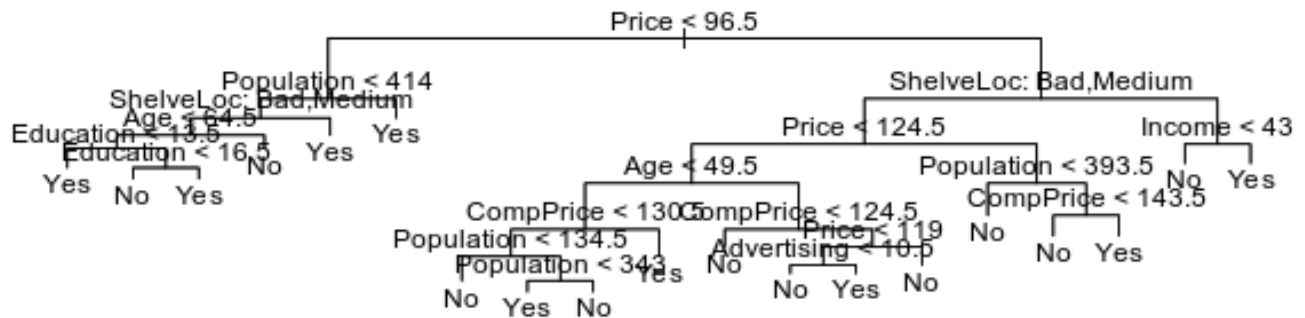
```
par(mfrow = c(1,1))
prune.carseats <- prune.misclass(tree.carseats, best = 9)
plot(prune.carseats)
text(prune.carseats, pretty = 0)
```



```
tree.pred <- predict(prune.carseats, carseats.test, type = "class")
table(tree.pred, high.test)
```

```
      high.test
tree.pred No Yes
No      97  25
Yes     20  58
```

```
prune.carseats <- prune.misclass(tree.carseats, best = 15)
plot(prune.carseats)
text(prune.carseats, pretty = 0)
```



```
tree.pred <- predict(prune.carseats, carseats.test, type = "class")
table(tree.pred, high.test)
```

```
      high.test
tree.pred No Yes
      No  102  30
      Yes   15  53
```

## Regression Trees

```
boston <- as.data.table(Boston)

N <- nrow(boston)

set.seed(1)

train <- sample(1:N, N / 2)

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"
```

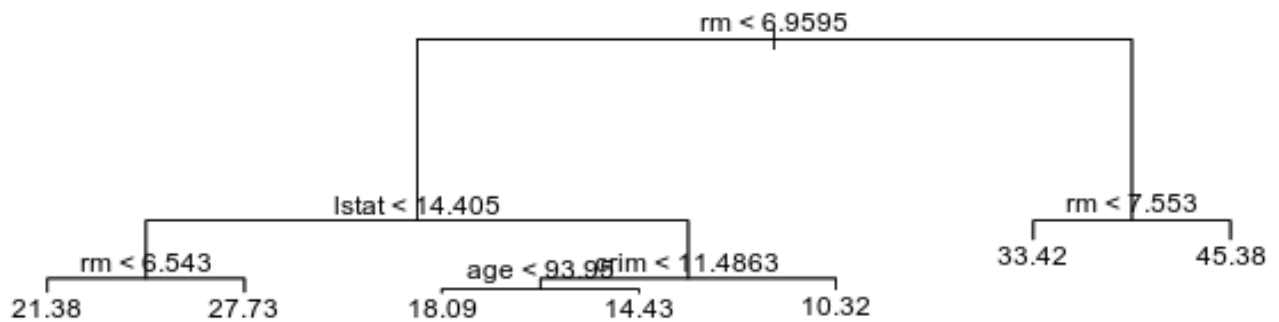
Number of terminal nodes: 7

Residual mean deviance: 10.38 = 2555 / 246

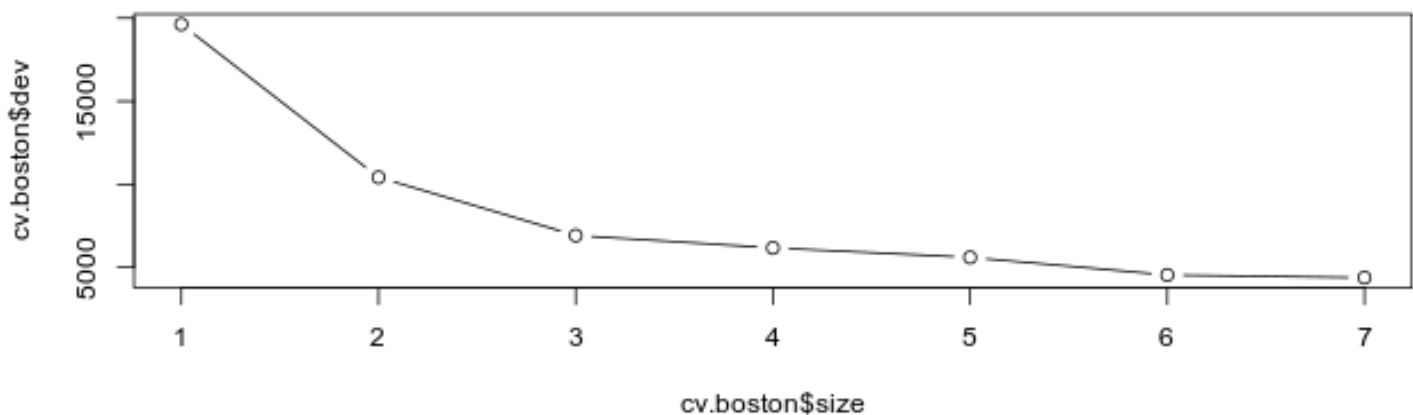
Distribution of residuals:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-10.1800	-1.7770	-0.1775	0.0000	1.9230	16.5800

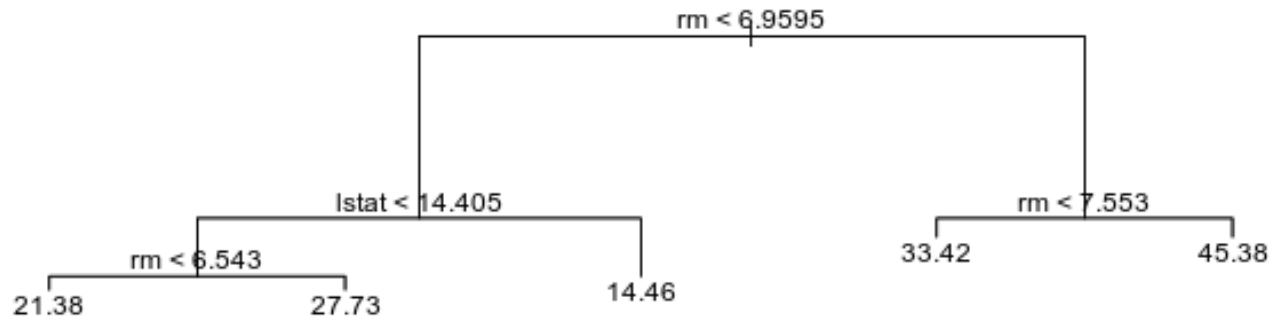
```
plot(tree.boston)
text(tree.boston, pretty = 0)
```



```
cv.boston <- cv.tree(tree.boston)
plot(cv.boston$size, cv.boston$dev, type = 'b')
```

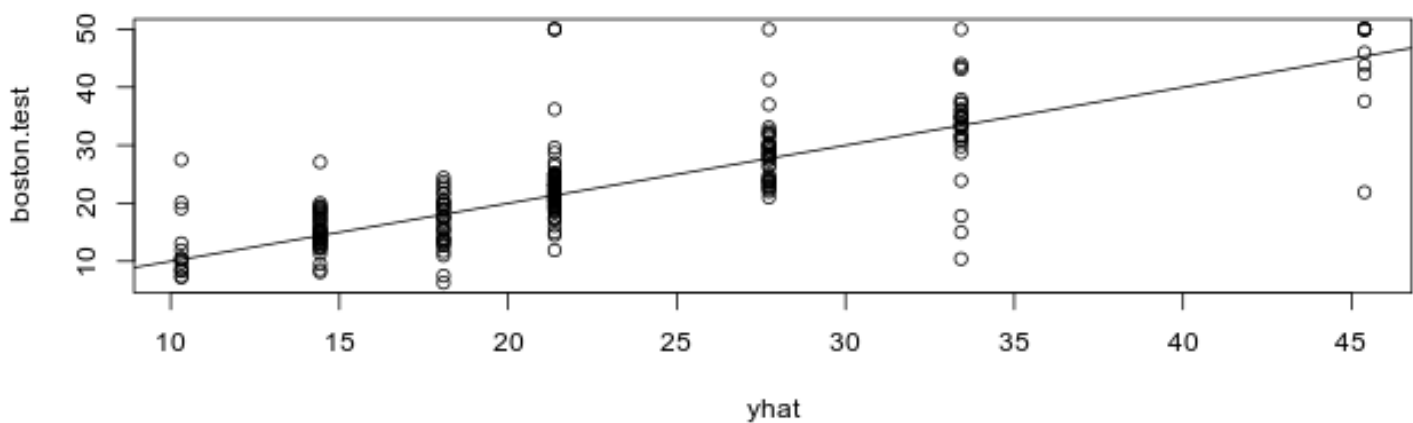


```
prune.boston <- prune.tree(tree.boston, best = 5)
plot(prune.boston)
text(prune.boston, pretty = 0)
```



```
yhat <- predict(tree.boston, newdata = boston[-train,])
boston.test <- boston[-train]$medv

plot(yhat, boston.test)
abline(0, 1)
```



```
mean((yhat - boston.test)^2)
```

```
[1] 35.28688
```

## Bagging and Boosting

```
set.seed(1)
```

```
bag.boston <- randomForest(medv ~ ., data = boston, subset = train, mtry = 13, importance = T)
bag.boston
```

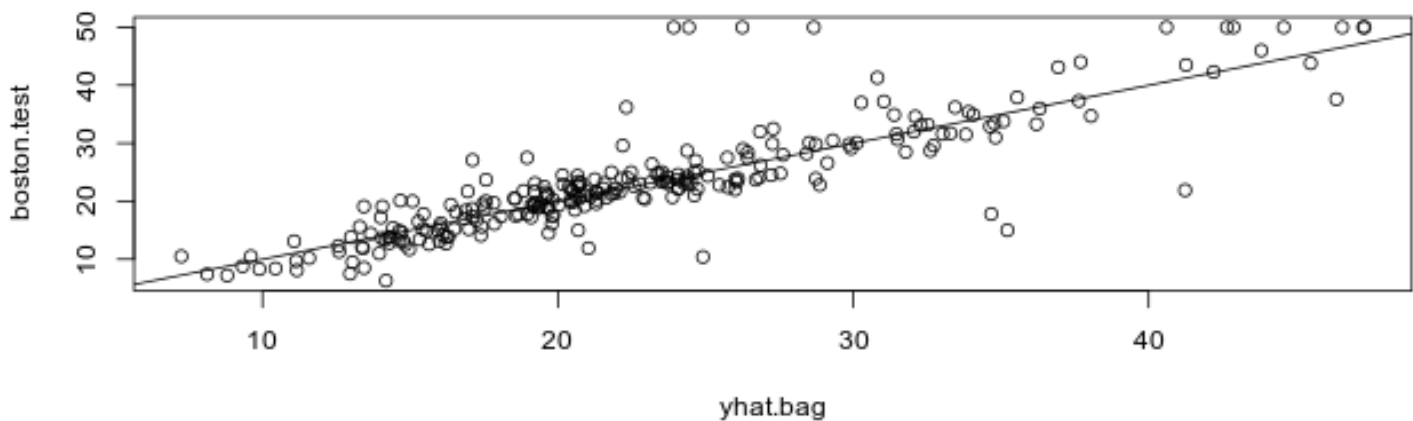
Call:

```
randomForest(formula = medv ~ ., data = boston, mtry = 13, importance = T,      subset = train
              Type of random forest: regression
              Number of trees: 500
```

```
No. of variables tried at each split: 13
```

```
Mean of squared residuals: 11.39601
% Var explained: 85.17
```

```
yhat.bag <- predict(bag.boston, newdata = boston[-train])
plot(yhat.bag, boston.test)
abline(0, 1)
```



```
mean((yhat.bag - boston.test)^2)
```

```
[1] 23.59273
```

```
bag.boston <- randomForest(medv ~ ., data = boston, subset = train, mtry = 13, ntree = 25)
yhat.bag <- predict(bag.boston, newdata = boston[-train,])
mean((yhat.bag - boston.test)^2)
```

```
[1] 23.66716
```

```
set.seed(1)
```

```
rf.boston <- randomForest(medv ~ ., data = boston, subset = train, mtry = 6, importance = T)
yhat.rf <- predict(rf.boston, newdata = boston[-train,])
mean((yhat.rf-boston.test)^2)
```

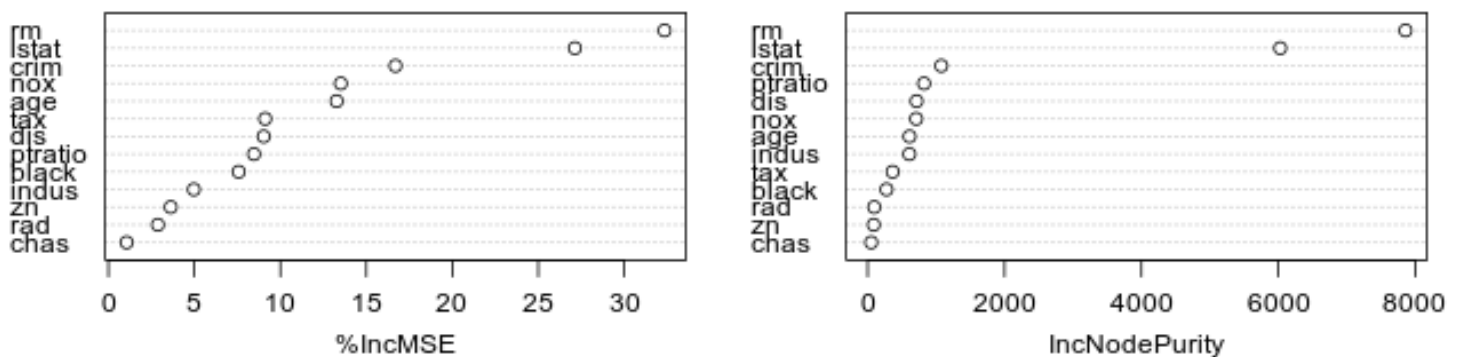
```
[1] 19.62021
```

```
importance(rf.boston)
```

	%IncMSE	IncNodePurity
crim	16.697017	1076.08786
zn	3.625784	88.35342
indus	4.968621	609.53356
chas	1.061432	52.21793
nox	13.518179	709.87339
rm	32.343305	7857.65451
age	13.272498	612.21424
dis	9.032477	714.94674
rad	2.878434	95.80598
tax	9.118801	364.92479
ptratio	8.467062	823.93341
black	7.579482	275.62272
lstat	27.129817	6027.63740

```
varImpPlot(rf.boston)
```

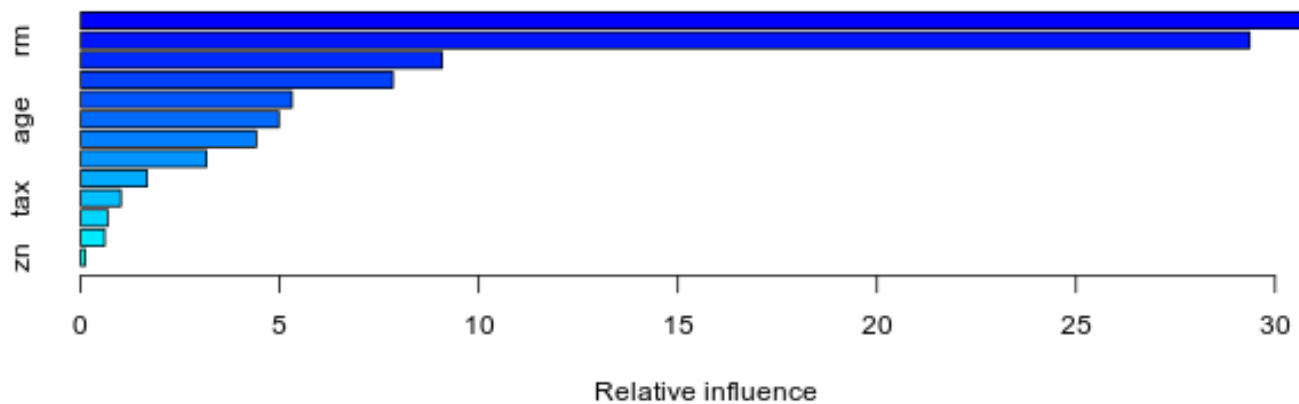
rf.boston





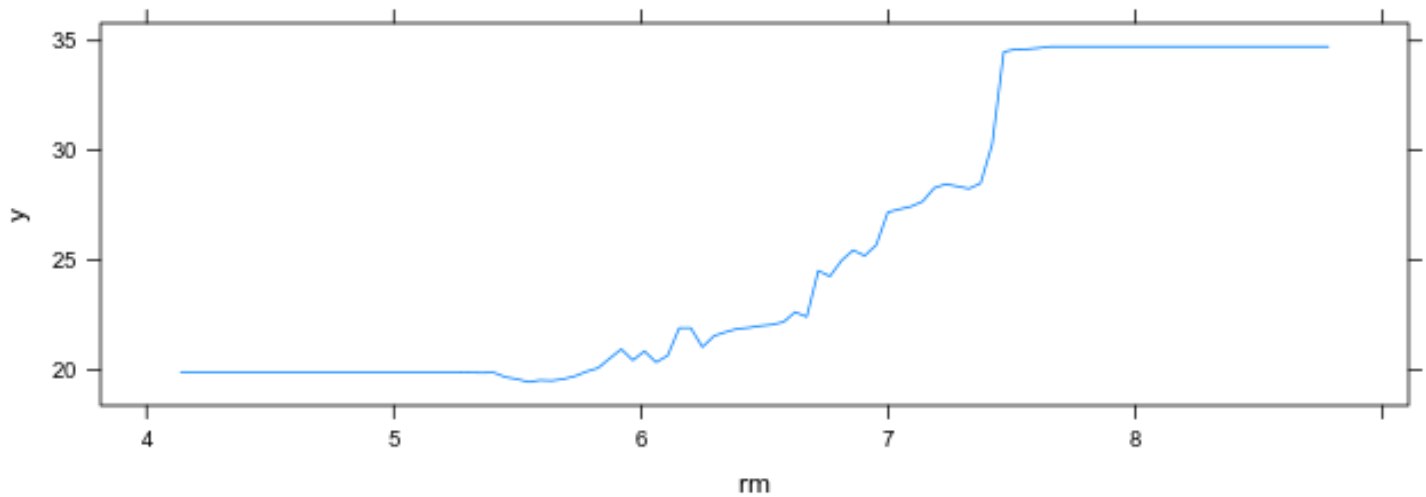
## Boosting

```
boost.boston <- gbm(medv ~ ., data = boston[-train,], distribution = "gaussian", n.trees = 5000)
summary(boost.boston)
```

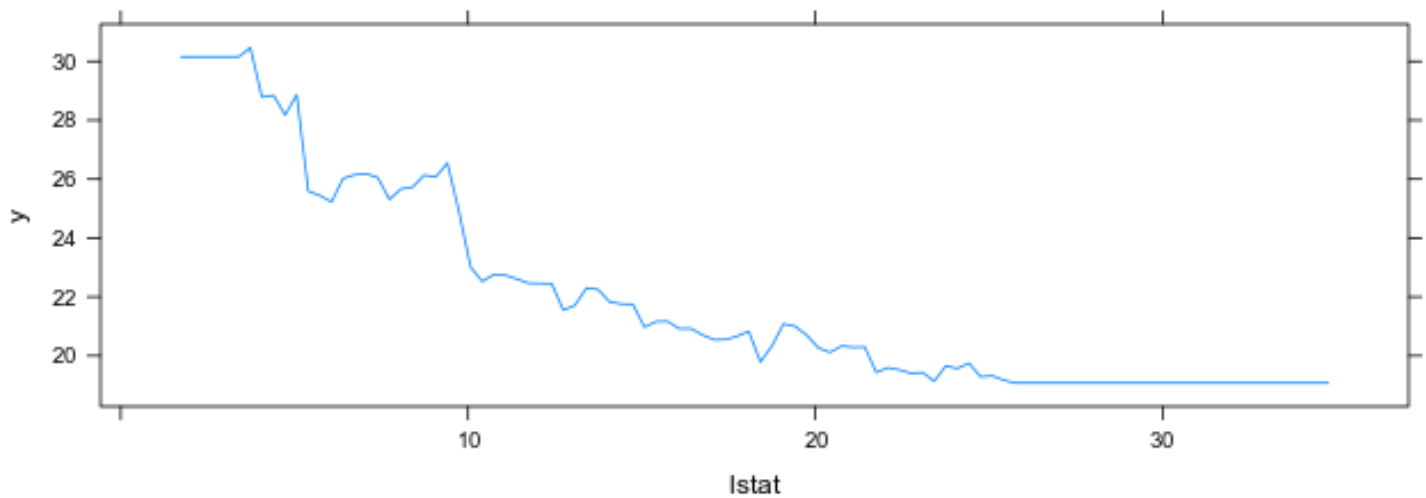


var	rel.inf
lstat	31.7768815
rm	29.3618406
dis	9.0801247
crim	7.8431133
nox	5.3077691
age	4.9799119
black	4.4118851
ptratio	3.1614540
indus	1.6642386
tax	1.0107120
chas	0.6850917
rad	0.6031076
zn	0.1138700

```
par(mfrow = c(1, 2))
plot(boost.boston, i = "rm")
```



```
plot(boost.boston, i = "lstat")
```



```
yhat.boost <- predict(boost.boston, newdata = boston[-train,], n.trees = 5000)
mean((yhat.boost - boston.test)^2)
```

```
[1] 2.242007e-06
```

## Applied

In the lab, we applied random forests to the *Boston* data using `mtry = 6` and using `ntree = 25` and `ntree = 500`. Create a plot displaying the test error resulting from random forests on this data set for a more comprehensive range of values for `mtry` `ntree`.

```

N <- nrow(boston)

train <- sample(1:N, N * .7)

boston.train <- boston[train]
boston.test <- boston[!train]

ntrees <- seq(0, 500, 10)

results <- numeric(length(ntrees))

for(i in 1:length(ntrees))
{
  tree.count <- ntrees[i]

  rf <- randomForest(medv ~ ., data = boston.train, mtry = 6, n.trees = tree.count)

  pred <- predict(rf, newdata = boston.test)

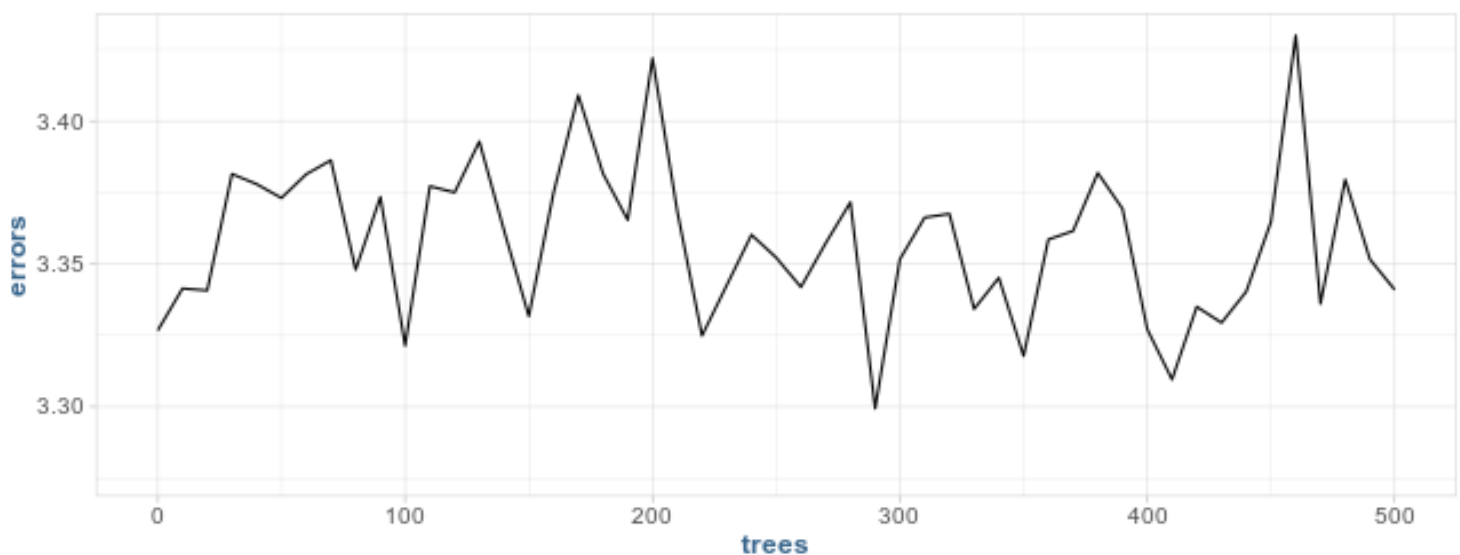
  results[i] <- sqrt(mean((pred - boston.test$medv)^2)) # store the rmse
}

lowest.error <- which.min(results)

rf.results <- data.table(trees = ntrees, errors = results)[, lowest := .I == lowest.error]

ggplot(rf.results, aes(trees, errors, fill = lowest)) +
  geom_line()

```



In the lab, a classification tree was applied to the Carseats data set after converting Sales into a qualitative response variable. Now we will seek to predict Sales using regression trees and related approaches, treating the response as a quantitative variable.

a.) Split the data into a training set and a test set.

```
N <- nrow(carseats)

index <- sample(1:N, N * .7)

carseats.train <- carseats[index]
carseats.test <- carseats[!index]
```

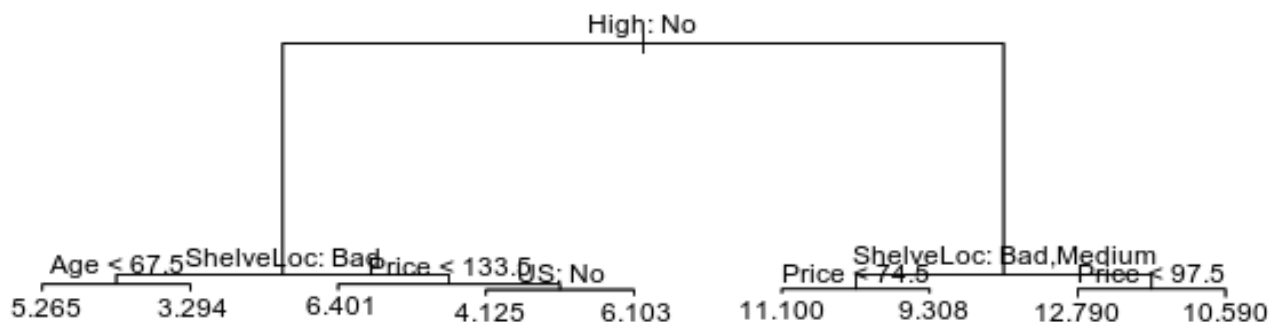
b.) Fit a regression tree to the training set. Plot the tree, and interpret the results. What test MSE do you obtain?

```
rf <- tree(Sales ~., data = carseats.train)
pred <- predict(rf, newdata = carseats.test)
mse <- mean((pred - carseats.test$Sales)^2)
```

```
mse
```

```
[1] 3.004485
```

```
par(mfrow = c(1,1))
plot(rf)
text(rf, pretty = 0)
```



c.) Use cross-validation in order to determine the optimal level of tree complexity. Does pruning the tree improve the test MSE?