Homework 8

Joshua Oswari - A14751270

4/30/2019

Problem 1

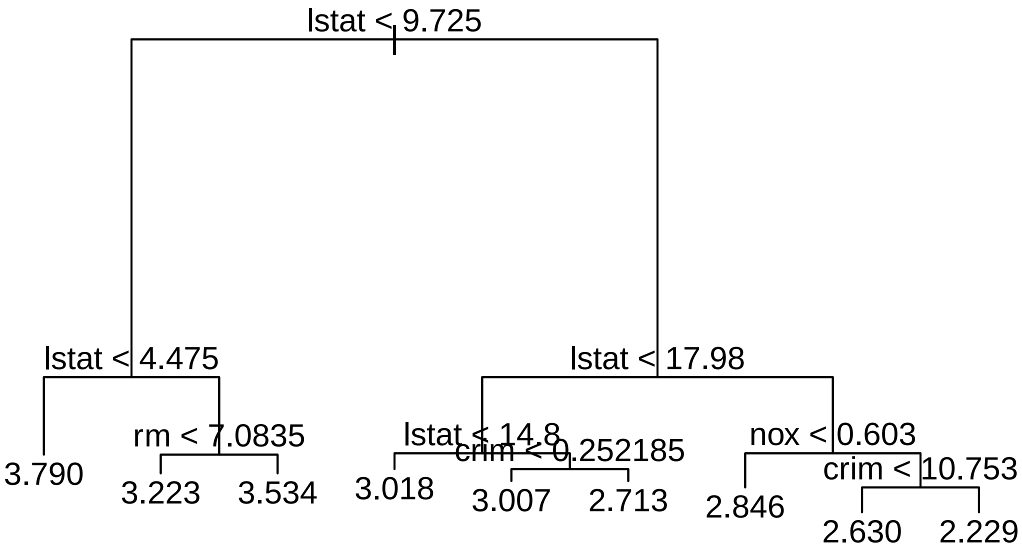
```
data = read.csv("~/Math189/boston.csv")

library(tree)
data = data[,2:15]
names(data)
```

```
##  [1] "crim"    "zn"      "indus"   "chas"    "nox"     "rm"      "age"
##  [8] "dis"     "rad"     "tax"     "ptratio" "bk"      "lstat"   "medv"
```

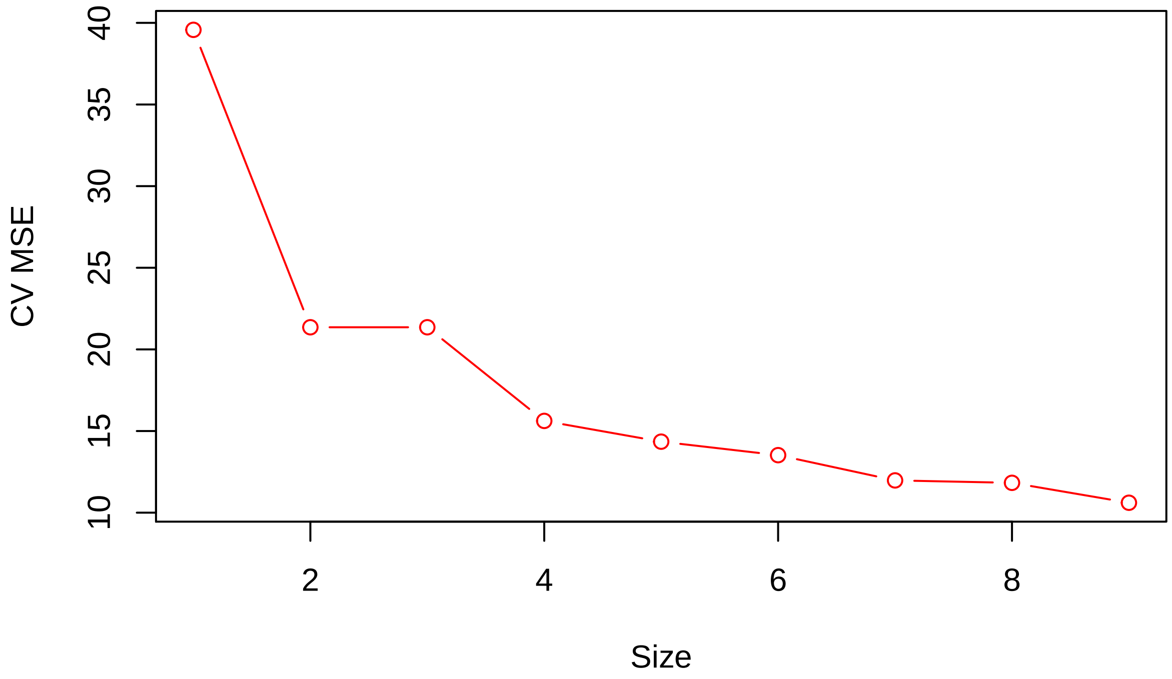
```
# Randomly divide the sample into training set and testing set
train = sample(1:nrow(data), (1/2)*nrow(data)+1)

# Train a regression tree
tree.data = tree(log(data$medv)~.,data,subset=train)
summary(tree.data)
```

```
##
## Regression tree:
## tree(formula = log(data$medv) ~ ., data = data, subset = train)
## Variables actually used in tree construction:
## [1] "lstat" "rm"    "crim"  "nox"
## Number of terminal nodes:  9
## Residual mean deviance:  0.02591 = 6.349 / 245
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.538100 -0.087800 -0.006021  0.000000  0.094740  0.688900
```

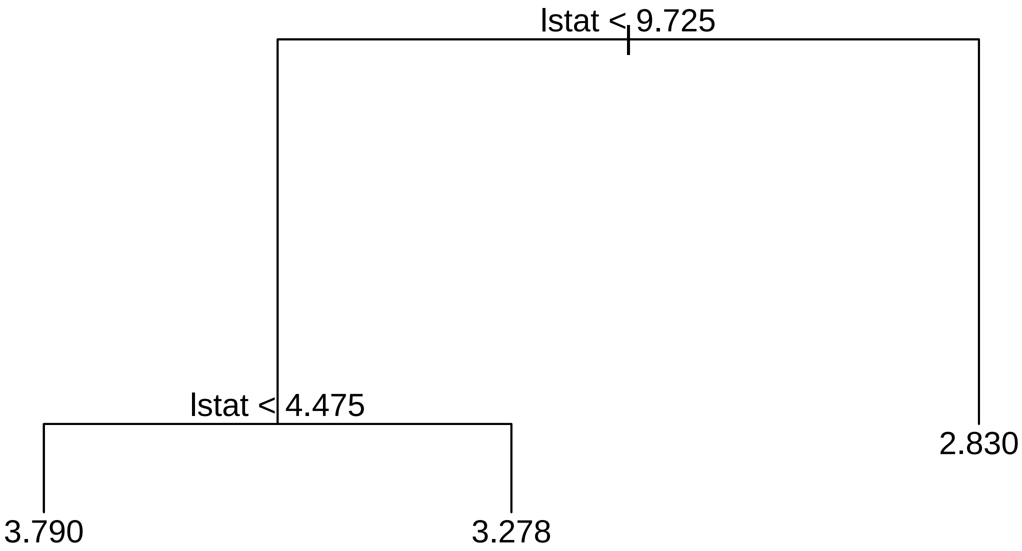
```
# Plot the generated tree

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



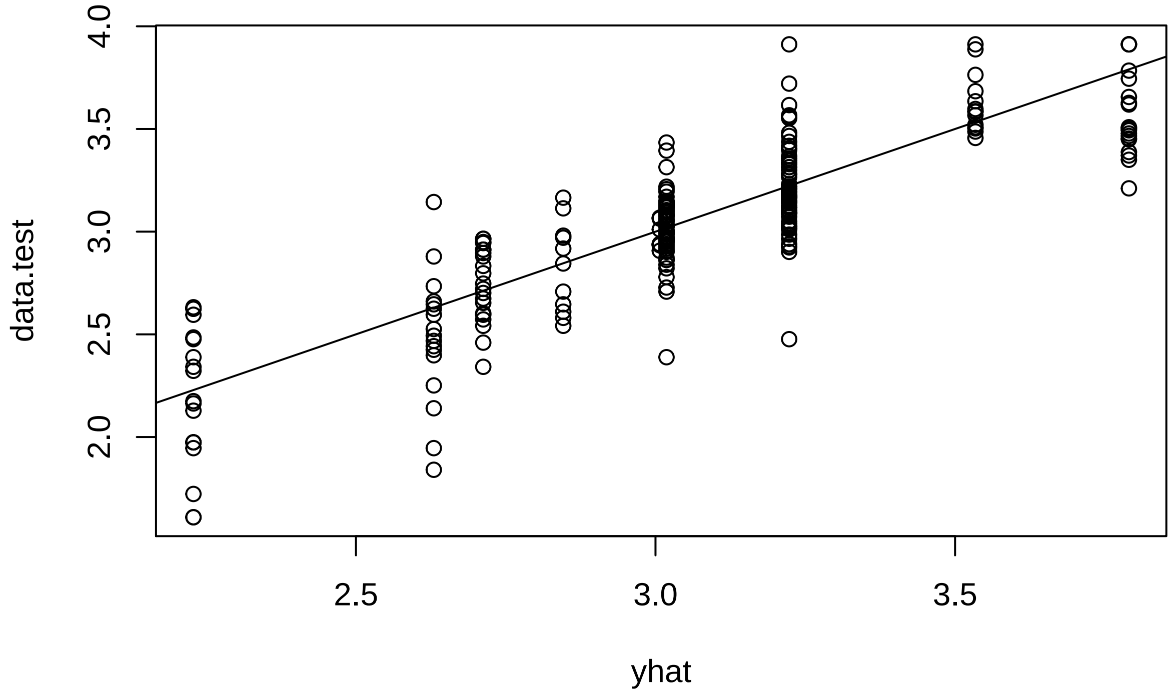
```
# Use the cv.tree() function to see whether pruning the tree will improve performance.
cv.data =cv.tree(tree.data,K=5)
plot(cv.data$size, cv.data$dev, type="b", xlab="Size", ylab="CV MSE", col="red")
```



```r
cv.size = cv.data$size[which.min(cv.data$dev)]

# Use the cv selected tree size to prune the tree
prune.data = prune.tree(tree.data, best=3)
plot(prune.data)
text(prune.data, pretty=0)
```



```
# Calculate prediction error on the test set
yhat = predict(tree.data, newdata=data[-train ,])
data.test = log(data[-train,"medv"])

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



```
# MSE on testing set
MSE.tree=mean((yhat-data.test)^2)
```

Problem 2

```
##################
#### Bagging  ####
##################

library(randomForest)
```

```
## randomForest 4.6-14
```

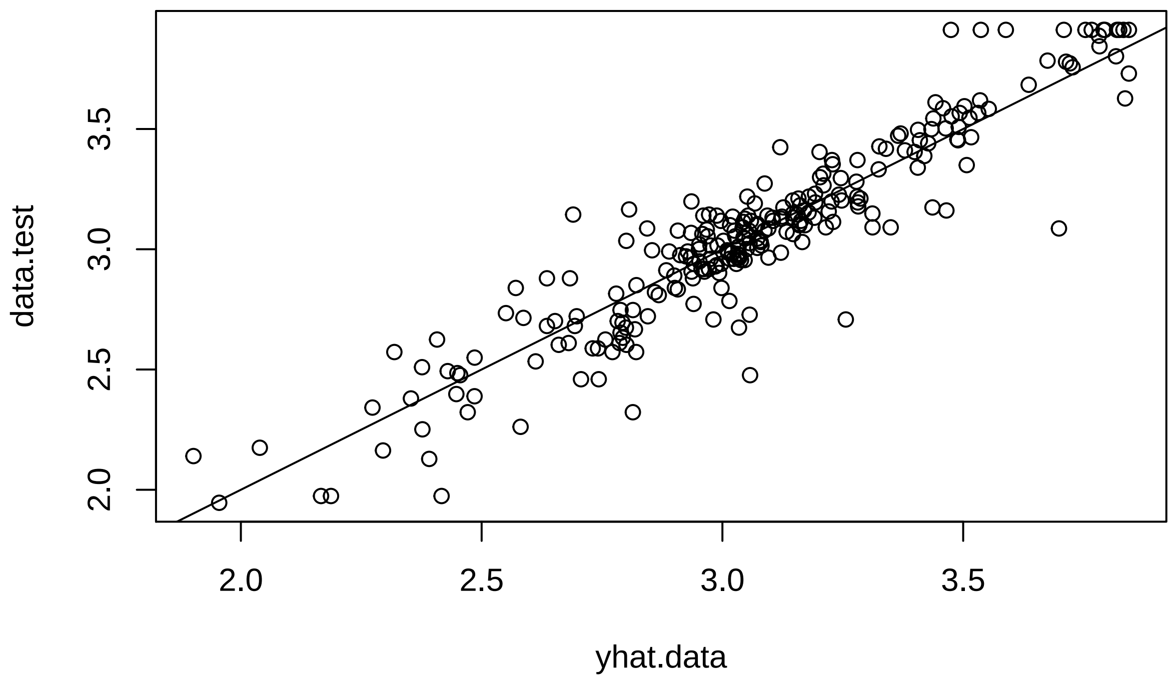
```
## Type rfNews() to see new features/changes/bug fixes.
```

```
train = sample(1:nrow(data), (1/2)*nrow(data)+1)

# Bagging (Random forests with m=p) and 100 trees
bag.data = randomForest(log(data$medv)~., data=data, subset=train,
                           mtry=(ncol(data)-1), importance=TRUE, ntree=100)
bag.data
```

```
##
## Call:
```

```
##  randomForest(formula = log(data$medv) ~ ., data = data, mtry = (ncol(data) -      1), importance = T
##                 Type of random forest: regression
##                       Number of trees: 100
## No. of variables tried at each split: 13
##
##           Mean of squared residuals: 0.02927946
##                     % Var explained: 82.42
```

```
# Prediction on test set
yhat.data = predict(bag.data, newdata=data[-train ,])
data.test = log(data[-train,"medv"])

# Plot prediction performance
plot(yhat.data, data.test)
abline (0,1)
```

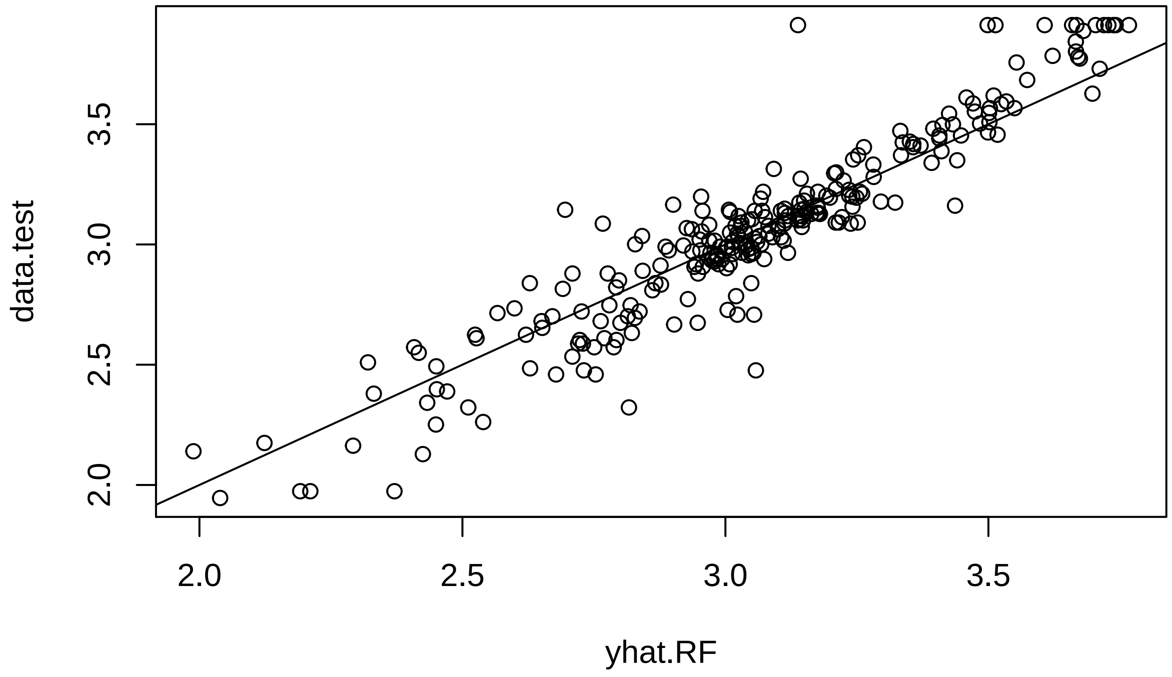


```
# MSE on test set
MSE.bag=mean((yhat.data-data.test)^2)

#########################
#### Random Forests  ####
#########################

library(randomForest)
```

```
# Random forests with m=sqrt(p) and 100 trees
RF.data =randomForest(log(data$medv)~.,data=data,subset=train,
                          mtry=4, importance =TRUE, ntree=100)
RF.data
```

```
##
## Call:
##  randomForest(formula = log(data$medv) ~ ., data = data, mtry = 4,      importance = TRUE, ntree = 1
##                Type of random forest: regression
##                      Number of trees: 100
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 0.02675961
##                     % Var explained: 83.93
```

```
# Prediction on test set
yhat.RF = predict(RF.data, newdata=data[-train,])

# Plot prediction performance
plot(yhat.RF, data.test)
abline(0,1)
```



```
# MSE on test set
MSE.RF=mean((yhat.RF -data.test)^2)
```