Random Forests

Tahani Omer

12/16/2019

## Packages required

We will Load required packages,and have a look at the “Boston” data from “MASS” packages.

library(MASS)  
library(ISLR)  
library(tree)  
  
names(Boston)

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

str(Boston)

## 'data.frame': 506 obs. of 14 variables:  
## $ crim : num 0.00632 0.02731 0.02729 0.03237 0.06905 ...  
## $ zn : num 18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...  
## $ indus : num 2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...  
## $ chas : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ nox : num 0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...  
## $ rm : num 6.58 6.42 7.18 7 7.15 ...  
## $ age : num 65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...  
## $ dis : num 4.09 4.97 4.97 6.06 6.06 ...  
## $ rad : int 1 2 2 3 3 3 5 5 5 5 ...  
## $ tax : num 296 242 242 222 222 222 311 311 311 311 ...  
## $ ptratio: num 15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...  
## $ black : num 397 397 393 395 397 ...  
## $ lstat : num 4.98 9.14 4.03 2.94 5.33 ...  
## $ medv : num 24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...

## Exploratory data analysis

#summary statistics  
attach(Boston)  
summary(Boston)

## crim zn indus chas   
## Min. : 0.00632 Min. : 0.00 Min. : 0.46 Min. :0.00000   
## 1st Qu.: 0.08204 1st Qu.: 0.00 1st Qu.: 5.19 1st Qu.:0.00000   
## Median : 0.25651 Median : 0.00 Median : 9.69 Median :0.00000   
## Mean : 3.61352 Mean : 11.36 Mean :11.14 Mean :0.06917   
## 3rd Qu.: 3.67708 3rd Qu.: 12.50 3rd Qu.:18.10 3rd Qu.:0.00000   
## Max. :88.97620 Max. :100.00 Max. :27.74 Max. :1.00000   
## nox rm age dis   
## Min. :0.3850 Min. :3.561 Min. : 2.90 Min. : 1.130   
## 1st Qu.:0.4490 1st Qu.:5.886 1st Qu.: 45.02 1st Qu.: 2.100   
## Median :0.5380 Median :6.208 Median : 77.50 Median : 3.207   
## Mean :0.5547 Mean :6.285 Mean : 68.57 Mean : 3.795   
## 3rd Qu.:0.6240 3rd Qu.:6.623 3rd Qu.: 94.08 3rd Qu.: 5.188   
## Max. :0.8710 Max. :8.780 Max. :100.00 Max. :12.127   
## rad tax ptratio black   
## Min. : 1.000 Min. :187.0 Min. :12.60 Min. : 0.32   
## 1st Qu.: 4.000 1st Qu.:279.0 1st Qu.:17.40 1st Qu.:375.38   
## Median : 5.000 Median :330.0 Median :19.05 Median :391.44   
## Mean : 9.549 Mean :408.2 Mean :18.46 Mean :356.67   
## 3rd Qu.:24.000 3rd Qu.:666.0 3rd Qu.:20.20 3rd Qu.:396.23   
## Max. :24.000 Max. :711.0 Max. :22.00 Max. :396.90   
## lstat medv   
## Min. : 1.73 Min. : 5.00   
## 1st Qu.: 6.95 1st Qu.:17.02   
## Median :11.36 Median :21.20   
## Mean :12.65 Mean :22.53   
## 3rd Qu.:16.95 3rd Qu.:25.00   
## Max. :37.97 Max. :50.00

#Check for missing values  
sum(is.na(Boston))

## [1] 0

## Fitting Regression Trees

## Split data set into 50:50 train and test data

Now we creat a training and test set (253,253) split of 506 observations, grow the tree on the training srt , and evaluate its perfomance on the test set.

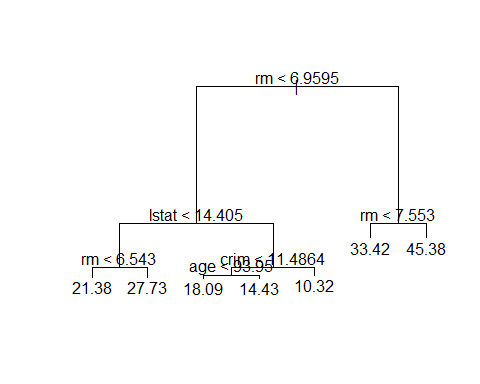
set.seed(1)  
train = sample(1:nrow(Boston), nrow(Boston)/2)  
tree.boston1=tree(medv~.,Boston,subset=train)  
summary(tree.boston1)

##   
## 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.boston1)  
text(tree.boston1,plot.new = plot(tree.boston1), pretty=0)

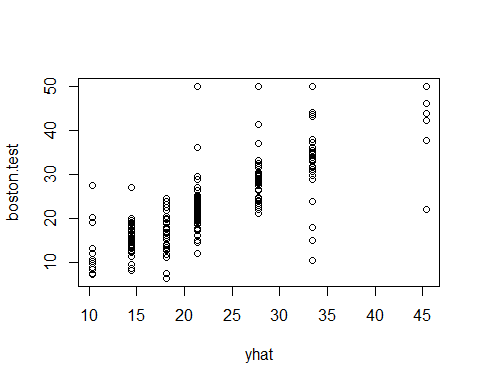
## Warning in text.default(xy$x[ind], xy$y[ind] + 0.5 \* charht, rows[ind], :  
## "plot.new" is not a graphical parameter

## Warning in text.default(xy$x[leaves], xy$y[leaves] - 0.5 \* charht, labels =  
## stat, : "plot.new" is not a graphical parameter



We will see how it performs in the test dataset.

yhat=predict(tree.boston1,newdata=Boston[-train,])  
boston.test=Boston[-train,"medv"]  
plot(yhat,boston.test)



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

## [1] 35.28688

Comments:

we get the mean square error of 35.28688 indicating that this model leads to test predictions that are within around $5.940276 of true median home value for the suburb

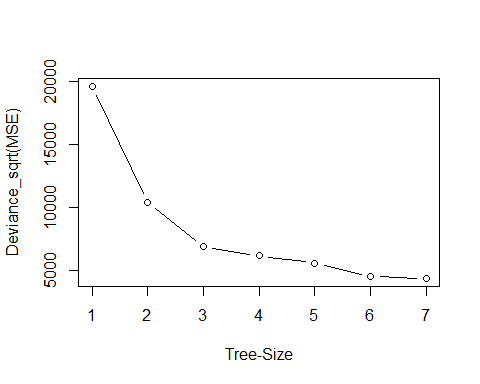
## Pruning

Now we use CV to prune the big tree, which might have too many variables, and print out the result.

cv.boston=cv.tree(tree.boston1)  
cv.boston

## $size  
## [1] 7 6 5 4 3 2 1  
##   
## $dev  
## [1] 4380.849 4544.815 5601.055 6171.917 6919.608 10419.472 19630.870  
##   
## $k  
## [1] -Inf 203.9641 637.2707 796.1207 1106.4931 3424.7810  
## [7] 10724.5951  
##   
## $method  
## [1] "deviance"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

plot(cv.boston$size,cv.boston$dev,type='b',xlab = "Tree-Size",ylab = "Deviance\_sqrt(MSE)")



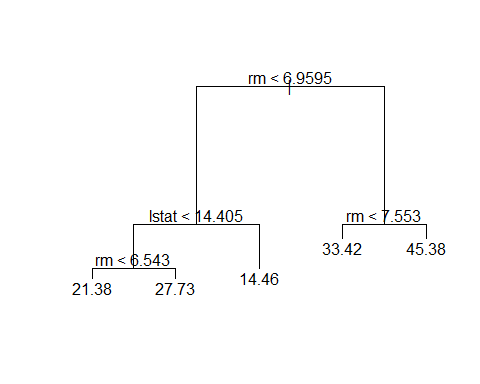
Comments: we got the size of the trees, and the deviance as the pruning proceeded. alse the cost complexity parameter from the plot we can pick the minimum value down near the minimum

Now we repeat the commands we had before

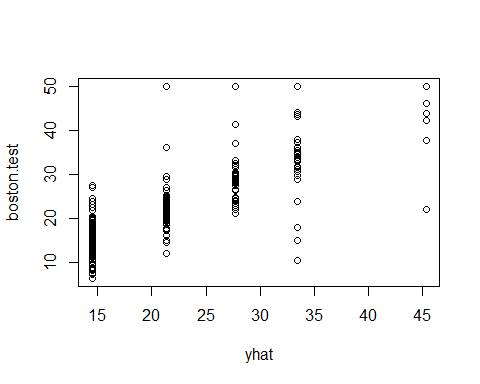
prune.boston=prune.tree(tree.boston1 ,best=5)  
plot(prune.boston)  
text(prune.boston,plot.new=plot(prune.boston), pretty =0)

## Warning in text.default(xy$x[ind], xy$y[ind] + 0.5 \* charht, rows[ind], :  
## "plot.new" is not a graphical parameter

## Warning in text.default(xy$x[leaves], xy$y[leaves] - 0.5 \* charht, labels =  
## stat, : "plot.new" is not a graphical parameter



yhat=predict(prune.boston,newdata=Boston[-train,])  
boston.test=Boston[-train,"medv"]  
plot(yhat,boston.test)



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

## [1] 35.90102

Comments: We did not get too much from pruning, except we get a small tree which is easier to interpret

## Random Forests

Random forests build lots of bushy trees, and then avaerage them to reduce the variance

library(randomForest)

## randomForest 4.6-14

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

library(MASS)  
set.seed(101)  
dim(Boston)

## [1] 506 14

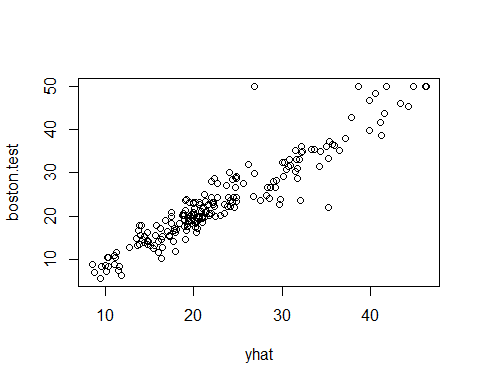
train = sample(1:nrow(Boston),300)

#we will use the respons 'medv',the median housing value(in\$1k dollars)  
  
rf.boston = randomForest(medv~.,data = Boston,subset = train)  
rf.boston

##   
## Call:  
## randomForest(formula = medv ~ ., data = Boston, subset = train)   
## Type of random forest: regression  
## Number of trees: 500  
## No. of variables tried at each split: 4  
##   
## Mean of squared residuals: 12.68651  
## % Var explained: 83.45

comments: The MSR and % variance explaind are based on OOB or-out-of-bage estimates. The model reports that ‘mtry=4’,which is the number of variable randomly chosen at each split.

yhat=predict(rf.boston,newdata=Boston[-train,])  
boston.test=Boston[-train,"medv"]  
plot(yhat,boston.test)



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

## [1] 11.41831

Comments:

we get the mean square error of 11.41831 indicating that this model leads to test predictions that are within around $3.379099 of true median home value for the suburb