Bagging, Random Forest and Boosting

Lab 12 - MATH 4322

- We will apply bagging, random forests and boosting to the `Boston` data, using the `randomForest` package.
- *Note*: The exact results obtained in this lab may depend on the version of `R` and the version of the `randomForest` package installed on your computer. Give the results from your computer.
- You can use the `Rmarkdown` script given or write down your answers and scan them as a pdf file to upload in Canvas similar to your homework.
- Possible points: 10.

Question 1: For any data that has p predictors **bagging** requires that we consider how many predictors at each split in a tree?

Bagging will use all p predictors in a tree

First, we call the data and create training/testing sets.

```
library(ISLR2)
set.seed(1)
train = sample(1:nrow(Boston),nrow(Boston)/2)
boston.test = Boston[-train,"medv"]
```

Bagging

We perform bagging as follows:

```
library(randomForest)
set.seed(10)
bag.boston = randomForest(medv~., data = Boston,
                          subset = train,
                          mtry = ncol(Boston) - 1,
                          importance = TRUE)
```

```r
bag.boston
```

```
Call:
 randomForest(formula = medv ~ ., data = Boston, mtry = ncol(Boston) -      1, importance = T
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 12

          Mean of squared residuals: 11.5691
                    % Var explained: 84.95
```

Question 2: What is the *MSE* based on the training set?

MSE = 11.5691

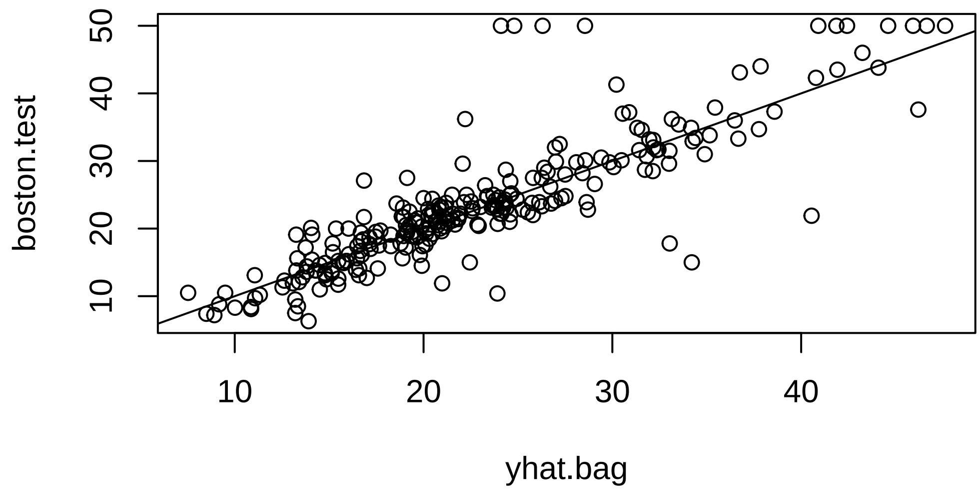
How well does this bagged model perform on the test set?

Question 3: What is the formula to determine the *MSE*?

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Run the following in `R`.

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




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

← Test MSE

```
[1] 23.23877
```

Question 4: What is the *MSE* of the test data set?

Test MSE = 23.23877

We could change the number of trees grown by `randomForest()` using the `ntree` argument:

```
bag.boston = randomForest(medv ~ ., data = Boston,
                          subset = train,
                          mtry = ncol(Boston) - 1,
                          ntree = 25)
bag.boston
```

```
Call:
 randomForest(formula = medv ~ ., data = Boston, mtry = ncol(Boston) -      1, ntree = 25, su
               Type of random forest: regression
                     Number of trees: 25
```

```
No. of variables tried at each split: 12

          Mean of squared residuals: 12.30361
                    % Var explained: 83.99
```

```
yhat.bag = predict(bag.boston,newdata = Boston[-train,])
mean((yhat.bag - boston.test)^2)
```

```
[1] 23.06258
```

Question 5: What method do we use to get the different trees?

We use bootstrap methods to get the different trees

Random Forests

Question 6: For a building a random forest of regression trees, what should be `mtry` (number of predictors to consider at each split)?

mtry = p/3 ← Regression mtry = $\sqrt{p}$ for classification

Type and run the following in R:

```
set.seed(10)
rf.boston = randomForest(medv ~., data = Boston,
                         subset = train,
                         mtry = (ncol(Boston)-1)/3,
                         importance = TRUE)
yhat.rf = predict(rf.boston,newdata = Boston[-train,])
mean((yhat.rf - boston.test)^2)
```

```
[1] 18.62328
```

Question 7: Compare the *MSE* of the test data to the *MSE* of the bagging.

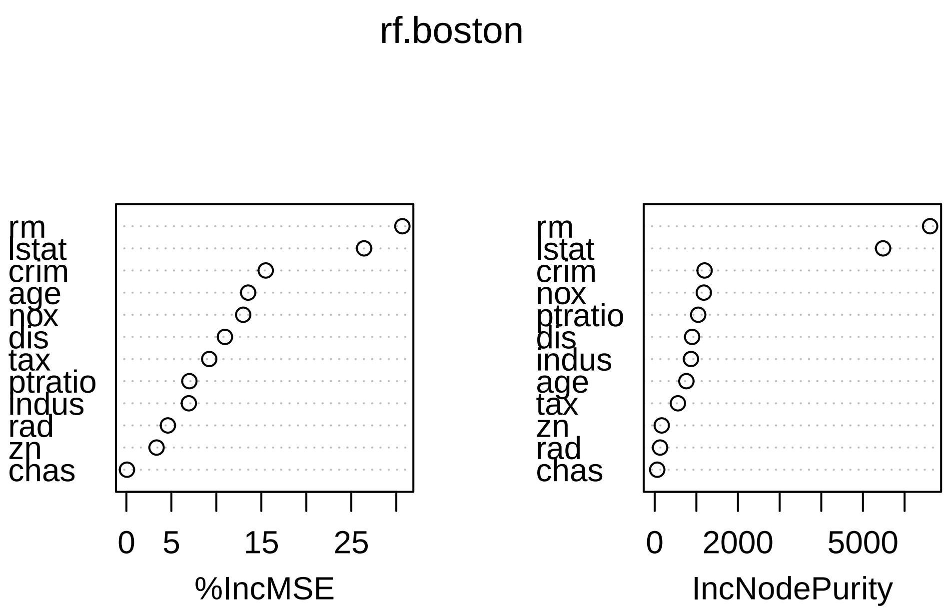
This MSE is smaller than the results of bagging.

Question 8: Use the `importance()` function what are the two most important variables?

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

```
              %IncMSE IncNodePurity
crim     15.48571304    1197.64717
zn        3.34978057     169.00931
indus     6.93488857     870.60348
chas      0.05746934      61.05778
nox      12.97835448    1179.66670
rm       30.67206810    6612.55554
age      13.52685213     760.41982
dis      10.94707995     899.17273
rad       4.60598124     129.80949
tax       9.20624202     556.89248
ptratio   6.99867017    1044.02812
lstat    26.41637352    5483.83696
```

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

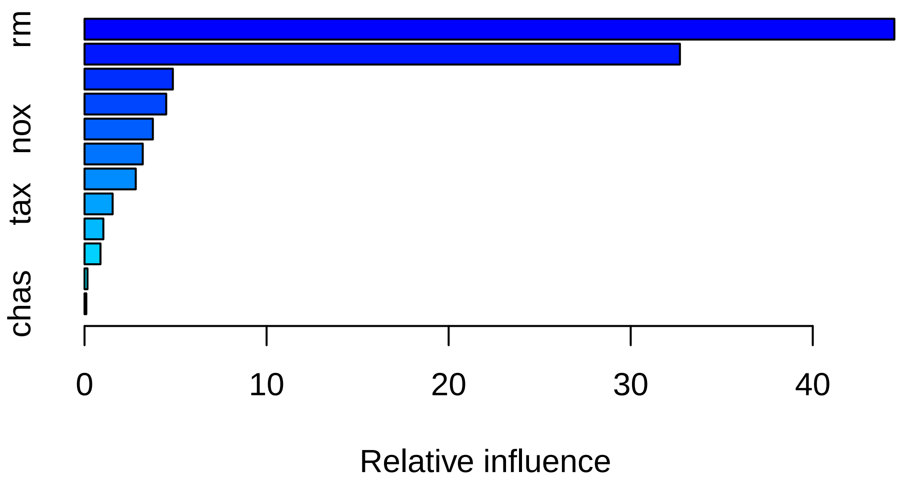


rm and lstat

Boosting

Run the following in R:

```r
library(gbm)
set.seed(1)
boost.boston = gbm(medv ~., data = Boston[train,],
                   distribution = "gaussian",
                   n.trees = 5000,
                   interaction.depth = 4)
summary(boost.boston)
```



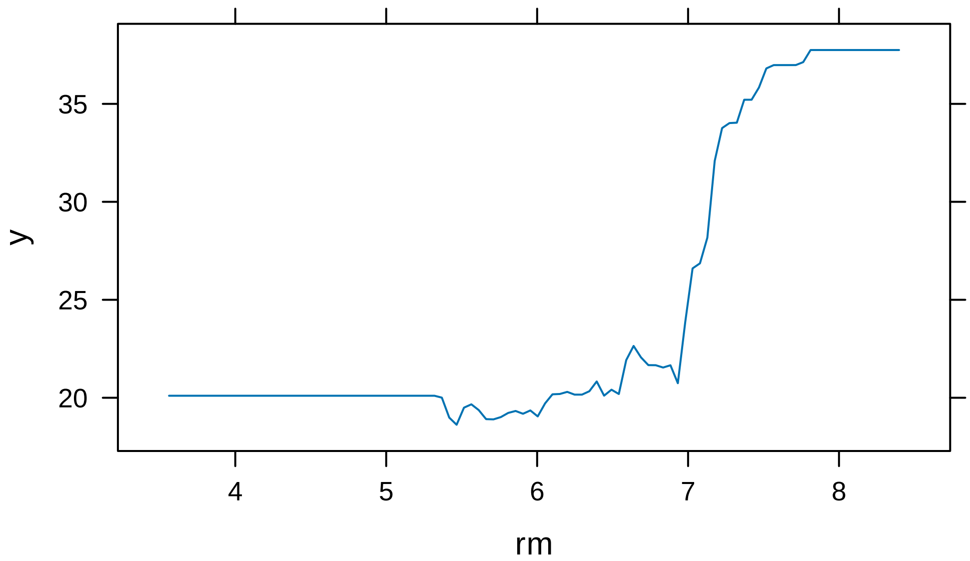

```
            var     rel.inf
rm           rm 44.48249588
lstat     lstat 32.70281223
crim       crim  4.85109954
dis         dis  4.48693083
nox         nox  3.75222394
age         age  3.19769210
ptratio ptratio  2.81354826
tax         tax  1.54417603
indus     indus  1.03384666
rad         rad  0.87625748
zn           zn  0.16220479
chas       chas  0.09671228
```

Question 9: What are the two most important variables with the boosted trees?

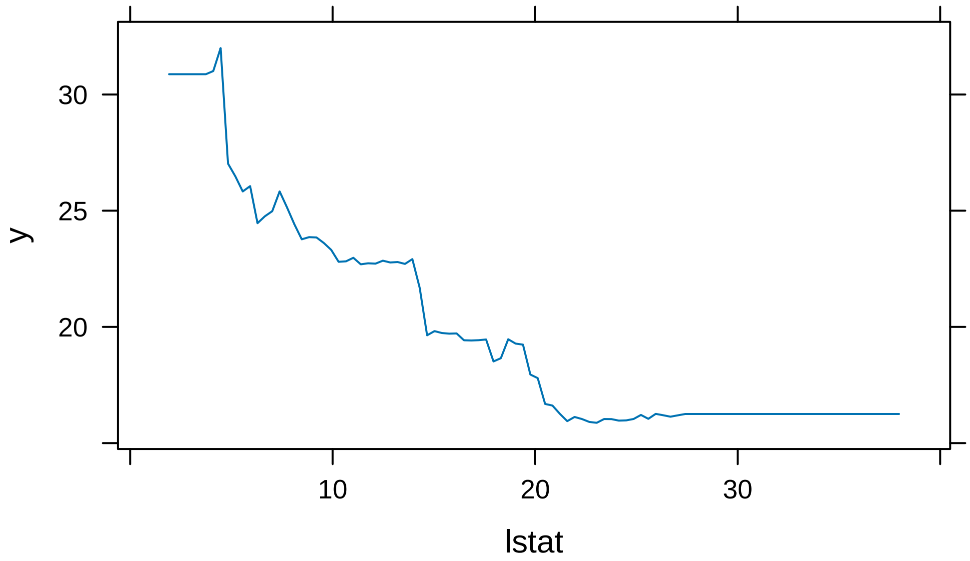
rm and lstat

We can produce *partial dependence plots* for these two variables. The plots illustrate the marginal effect of the selected variables on the response after *integrating* out the other variables.

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




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



Notice that the house prices are increasing with `rm` and decreasing with `lstat`.

We will use the boosted model to predict `medv` on the test set:

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

```
[1] 18.39057
```

Question 10: Compare this *MSE* to the *MSE* of the random forest and bagging models.

Bag MSE = 23.23877, RF MSE = 18.62328, Boost MSE = 18.39057