Lab 17 MATH 4322 Solutions

Bagging, Random Forest and Boosting

Fall 2022

- 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 BlackBoard 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? mtry = p

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

```
library(ISLR2)
```

```
## Warning: package 'ISLR2' was built under R version 4.2.1
```

```
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)
bag.boston
```

```
##
## Call:
##  randomForest(formula = medv ~ ., data = Boston, mtry = ncol(Boston) -      1, importance = TRUE, su
##                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
```

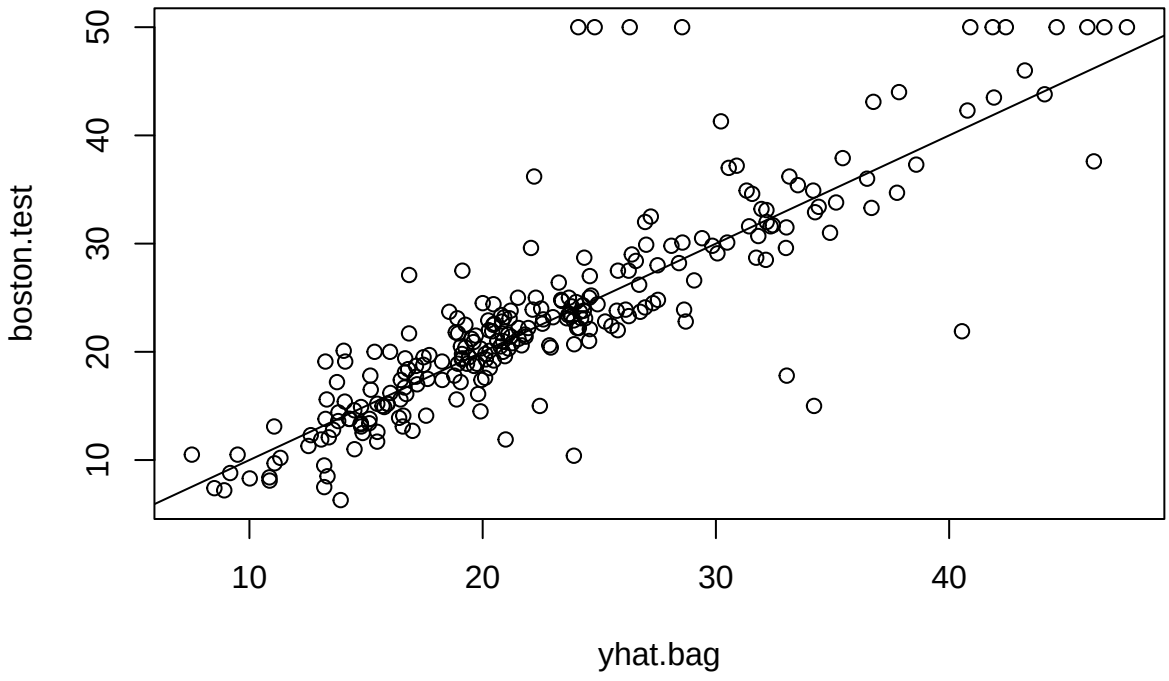
Question 2: What is the *MSE* based on the training set? This is 11.5691 but it is a squared value, this means we are off by $\sqrt{11.5691} = \$3.401338$ thousands

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

Question 3: What is the formula to determine the *MSE*?
MSE = mean(predicted y - observed y)^2

Run the following in `R`.

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




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

```
## [1] 23.23877
```

Question 4: What is the *MSE* of the test data set?
MSE = 23.23877 or $4.8206 thousands

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, subset = t
##                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?
This is the bootsrap aggregating

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)?
We use $p/3$ for regression tree and $\sqrt{p}$ for classification tree

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.
We get a lower MSE with the random forest

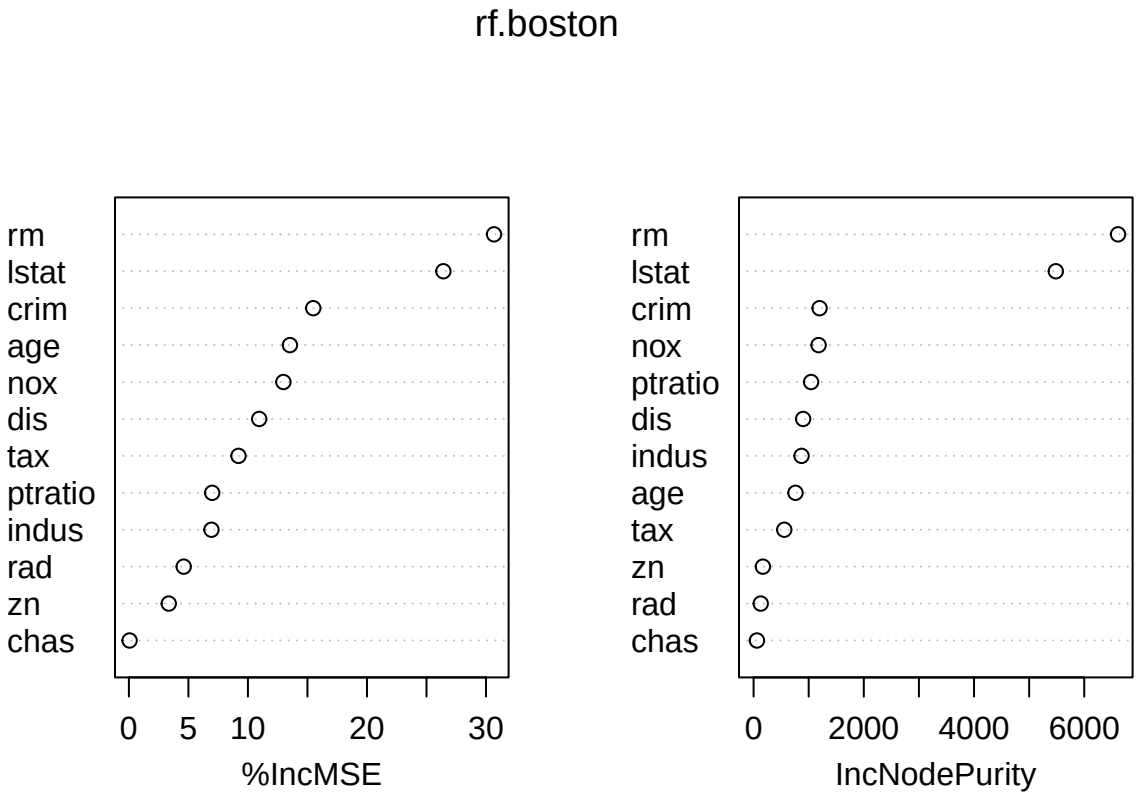
http://machinelearning202.pbworks.com/w/file/fetch/60606349/breiman_randomforests.pdf

Question 8: Use the `importance()` function what are the two mores 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)
```



'lstat' lower status of the populaiton and 'rm' average number of rooms per dwelling are the two most important variables

Boosting

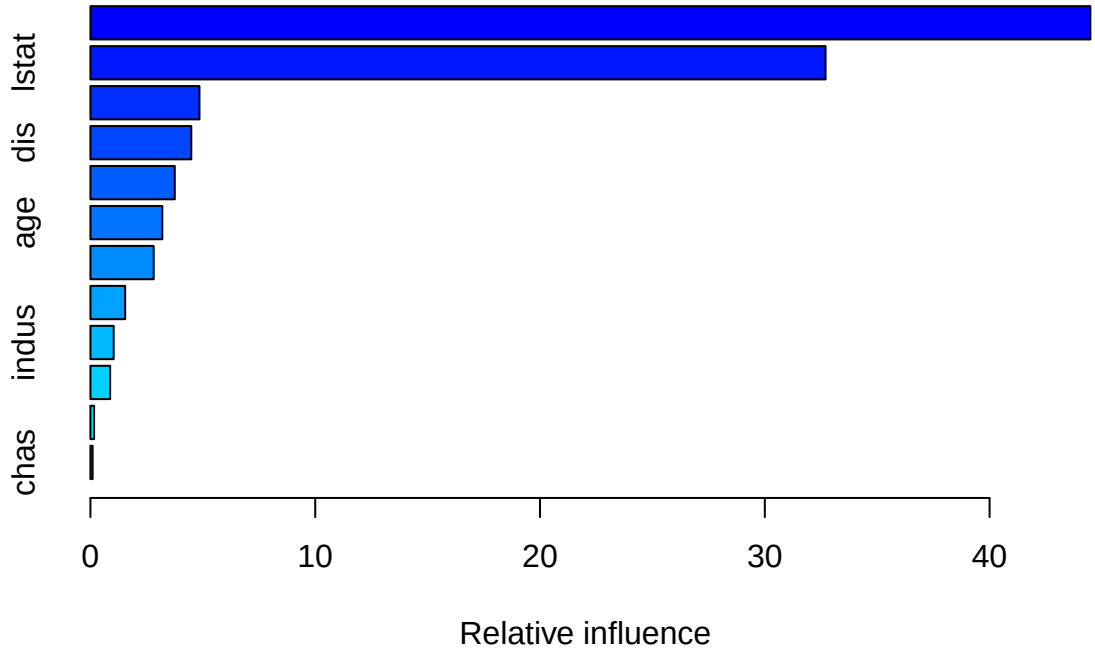
Run the following in `R`:

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.2.1
```

```
## Loaded gbm 2.1.8.1
```

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



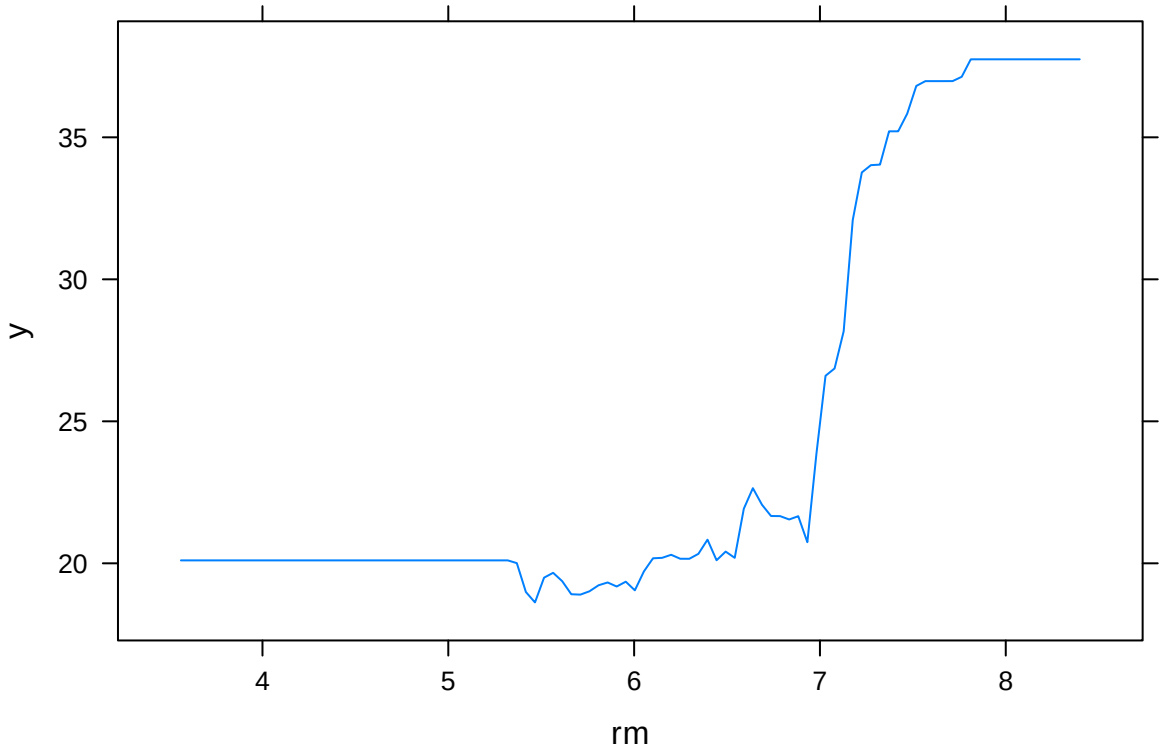

```
##             var     rel.inf
## rm           rm 44.48249588
## lstat     lstat 32.70281223
## crim       crim  4.85109954
## dis         dis  4.48693083
## nox         nox  3.75222394
## age         age  3.19769210
## ptratio ptratio  2.81354826
## tax         tax  1.54417603
## indus     indus  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?

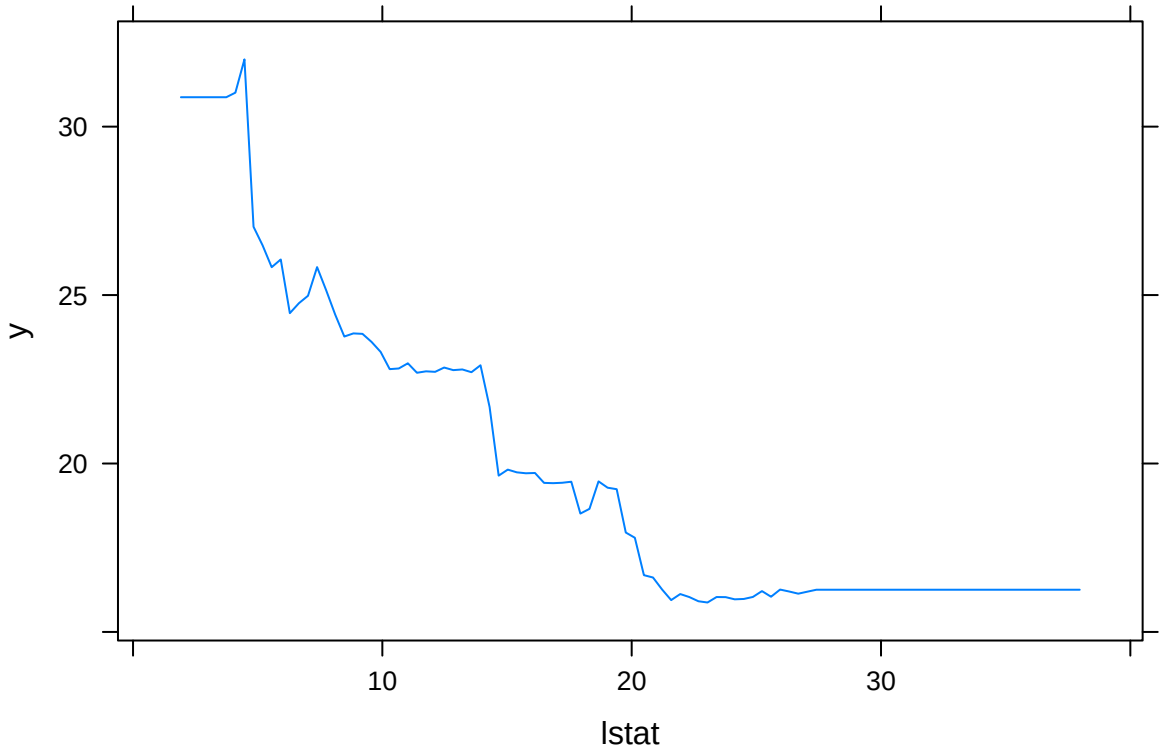
Similar as before ‘lstat‘ and ‘rm‘

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.
This is a little bit lower than the random forest.