Homework 5

Amin Baabol

Note: I collaborated with Mohamed Ahmed in the completion of the R program for part a of this recursive partitioning assignment. Our collaboration was limited to the plotting part of part a, specifically he helped understand how to do base R analogous ggplot of the decision tree and the observed vs. predicted median value ggplot in part a and a discussion about our conceptual comprehension of the various algorithms and models covered in chapter 9.

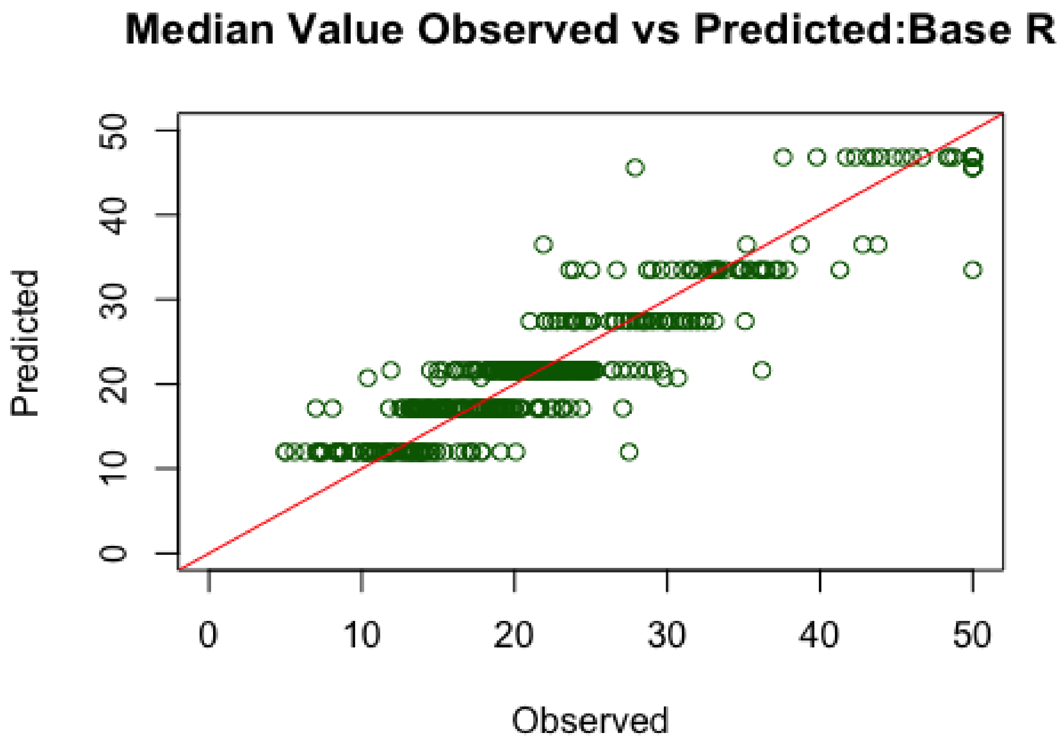
## Exercises

1. (Ex. 9.1 pg 186 in HSAUR, modified for clarity) The **BostonHousing** dataset reported by Harrison and Rubinfeld (1978) is available as a `data.frame` structure in the **mlbench** package (Leisch and Dimitriadou, 2009). The goal here is to predict the median value of owner-occupied homes (`medv` variable, in 1000s USD) based on other predictors in the dataset.

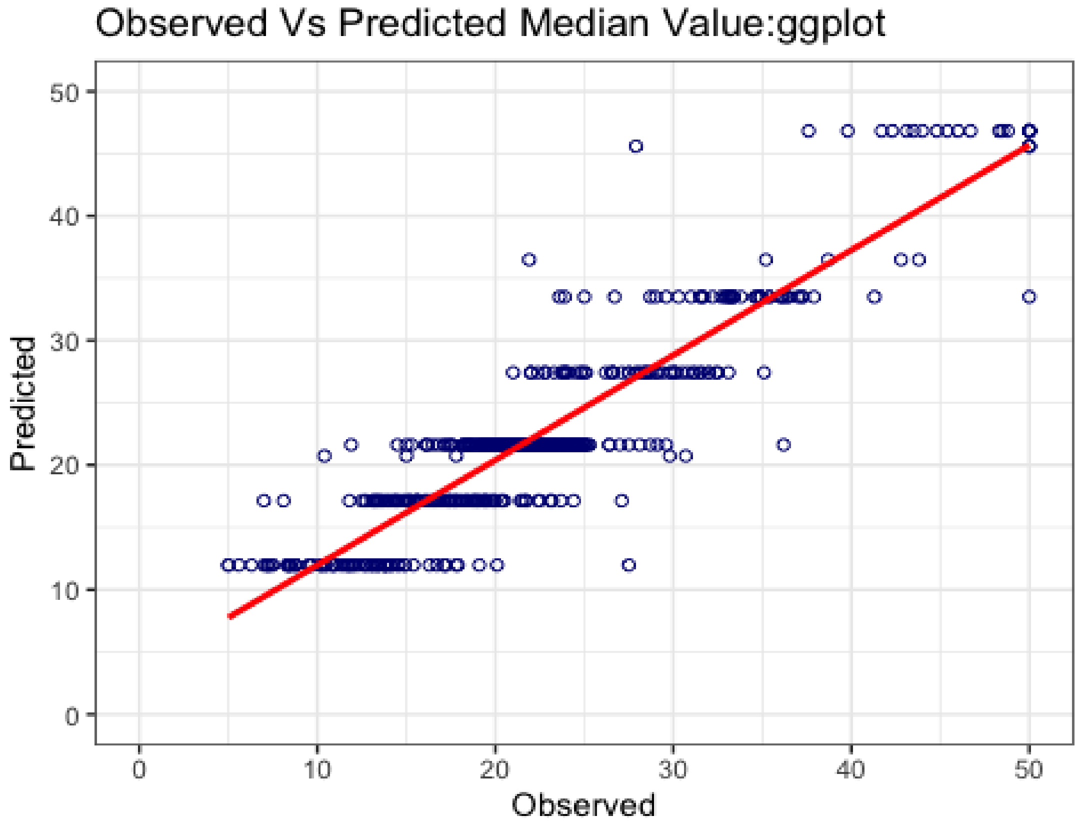
a) Construct a regression tree using rpart(). Discuss the results, including these key components:

```
##
## Regression tree:
## rpart(formula = medv ~ ., data = BostonHousing, control =
rpart.control(minsplit = 15))
##
## Variables actually used in tree construction:
## [1] crim    dis     lstat   ptratio rm
##
## Root node error: 42716/506 = 84.42
##
## n= 506
##
##         CP nsplit rel error  xerror     xstd
## 1 0.452744      0   1.00000 1.00447 0.083117
## 2 0.171172      1   0.54726 0.63498 0.054942
## 3 0.071658      2   0.37608 0.42494 0.044525
## 4 0.059002      3   0.30443 0.40421 0.043487
## 5 0.033756      4   0.24542 0.36334 0.042827
## 6 0.026613      5   0.21167 0.33431 0.039990
## 7 0.016986      6   0.18506 0.30543 0.039635
## 8 0.010429      7   0.16807 0.26212 0.034273
## 9 0.010000      8   0.15764 0.25926 0.033438

## n= 506
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
```

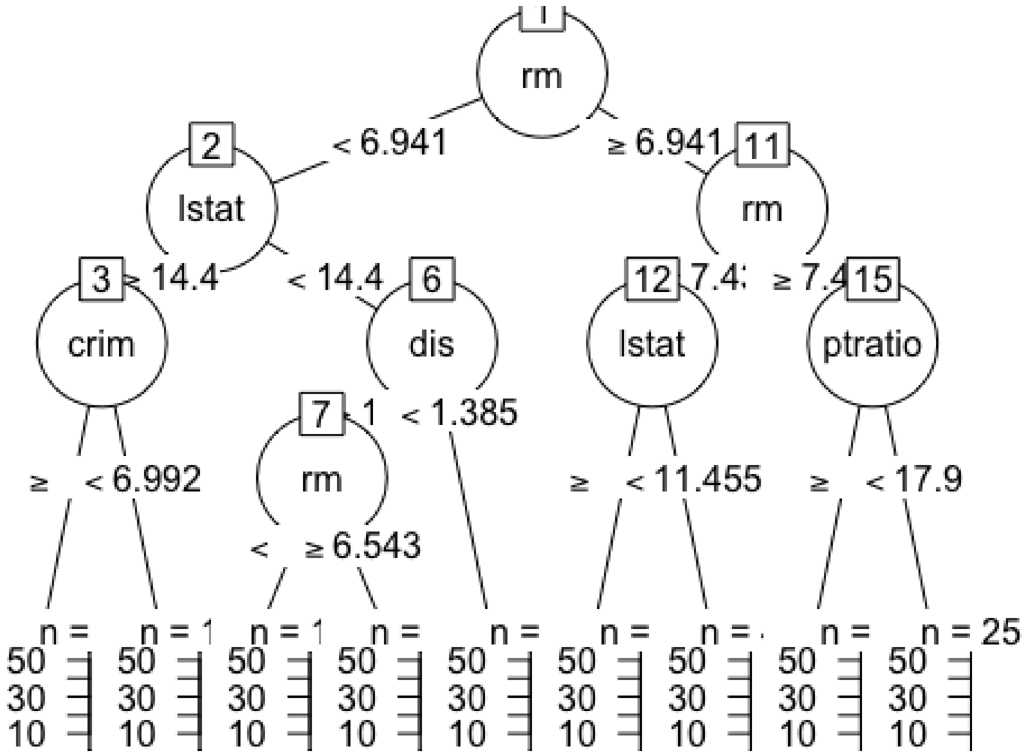
```
##  1) root 506 42716.3000 22.53281
##    2) rm< 6.941 430 17317.3200 19.93372
##      4) lstat>=14.4 175  3373.2510 14.95600
##        8) crim>=6.99237 74  1085.9050 11.97838 *
##        9) crim< 6.99237 101  1150.5370 17.13762 *
##      5) lstat< 14.4 255  6632.2170 23.34980
##       10) dis>=1.38485 250  3721.1630 22.90520
##         20) rm< 6.543 195  1636.0670 21.62974 *
##         21) rm>=6.543 55   643.1691 27.42727 *
##       11) dis< 1.38485 5   390.7280 45.58000 *
##    3) rm>=6.941 76  6059.4190 37.23816
##      6) rm< 7.437 46  1899.6120 32.11304
##       12) lstat>=11.455 5   329.7920 20.74000 *
##       13) lstat< 11.455 41   844.2200 33.50000 *
##      7) rm>=7.437 30  1098.8500 45.09667
##       14) ptratio>=17.9 5   312.6680 36.48000 *
##       15) ptratio< 17.9 25   340.7000 46.82000 *

## n= 506
## 
## node), split, n, deviance, yval
##       * denotes terminal node
## 
##  1) root 506 42716.3000 22.53281
##    2) rm< 6.941 430 17317.3200 19.93372
##      4) lstat>=14.4 175  3373.2510 14.95600
##        8) crim>=6.99237 74  1085.9050 11.97838 *
##        9) crim< 6.99237 101  1150.5370 17.13762 *
##      5) lstat< 14.4 255  6632.2170 23.34980
##       10) dis>=1.38485 250  3721.1630 22.90520
##         20) rm< 6.543 195  1636.0670 21.62974 *
##         21) rm>=6.543 55   643.1691 27.42727 *
##       11) dis< 1.38485 5   390.7280 45.58000 *
##    3) rm>=6.941 76  6059.4190 37.23816
##      6) rm< 7.437 46  1899.6120 32.11304
##       12) lstat>=11.455 5   329.7920 20.74000 *
##       13) lstat< 11.455 41   844.2200 33.50000 *
##      7) rm>=7.437 30  1098.8500 45.09667
##       14) ptratio>=17.9 5   312.6680 36.48000 *
##       15) ptratio< 17.9 25   340.7000 46.82000 *

## [1] 13.30788
```

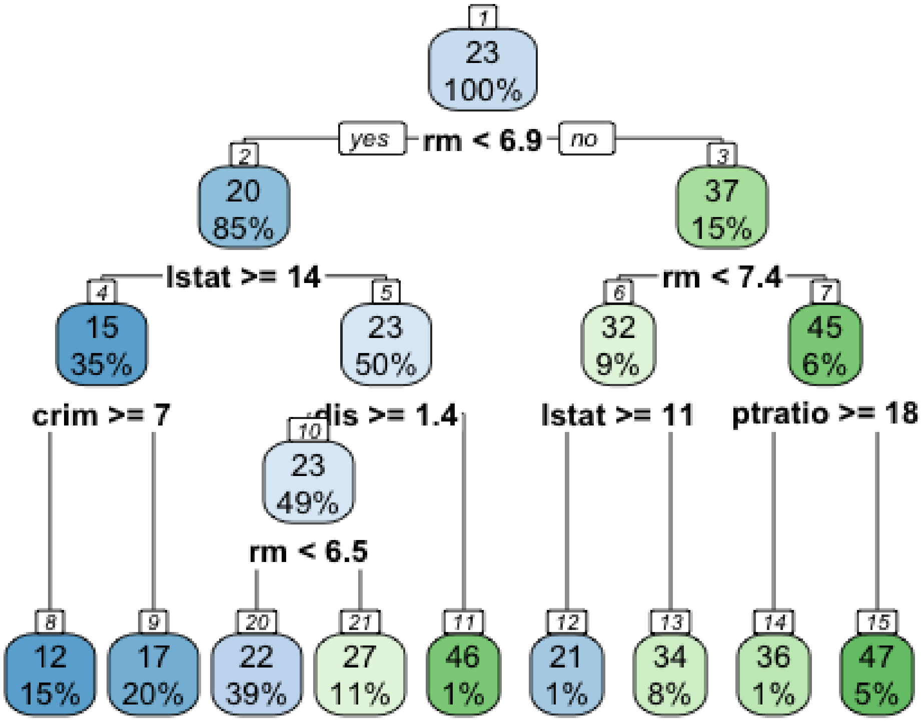


Median Value Observed vs Predicted:Base R



Observed Vs Predicted Median Value:ggplot


1
rm
2
< 6.941
≥ 6.941
11
lstat
rm
3
≥ 14.4
< 14.4
6
12
7.4:
≥ 7.4
15
crim
dis
lstat
ptratio
7
1
< 1.385
≥
< 6.992
rm
≥
< 11.455
≥
< 17.9
<
≥ 6.543
n =
n = 1
n = 1
n =
n =
n =
n =
n =
n = 25
50
30
10
50
30
10
50
30
10
50
30
10
50
30
10
50
30
10
50
30
10
50
30
10
50
30
10

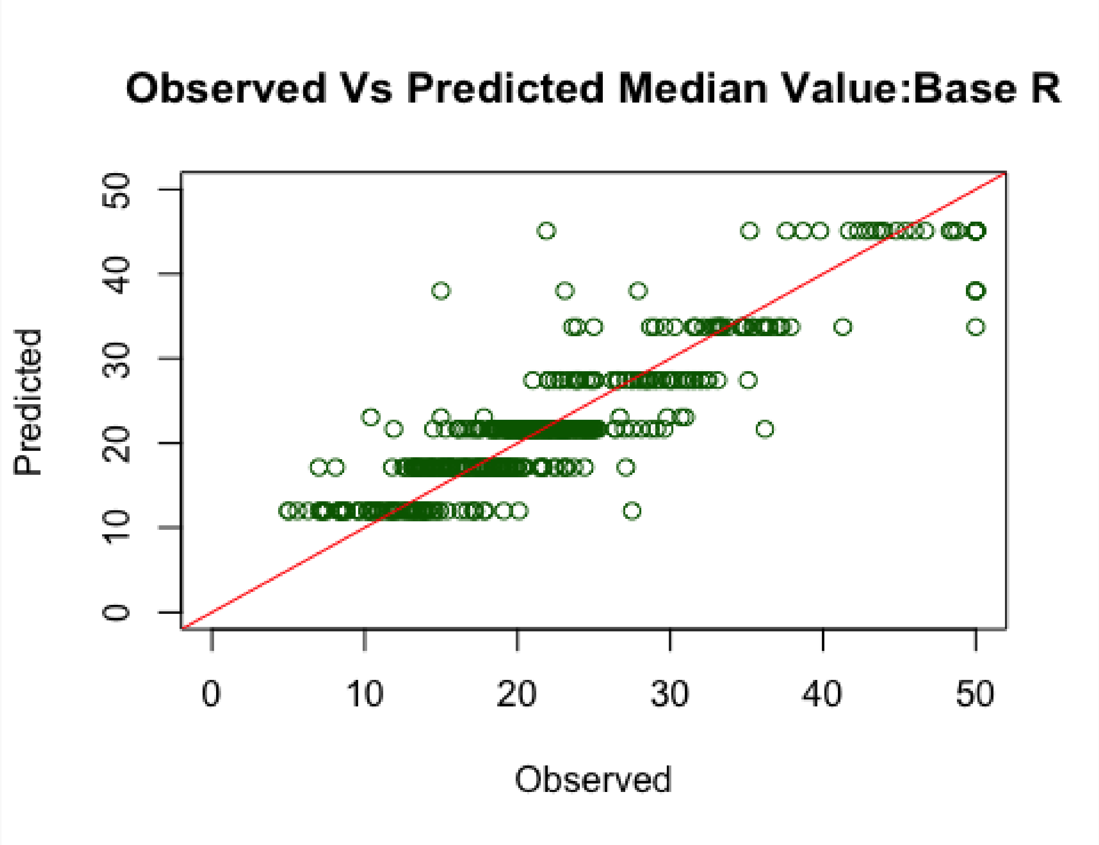

1
23
100%
yes
rm < 6.9
no
2
20
85%
3
37
15%
lstat >= 14
4
15
35%
5
23
50%
rm < 7.4
6
32
9%
7
45
6%
crim >= 7
dis >= 1.4
10
23
49%
lstat >= 11
ptratio >= 18
rm < 6.5
8
12
15%
9
17
20%
20
22
39%
21
27
11%
11
46
1%
12
21
1%
13
34
8%
14
36
1%
15
47
5%

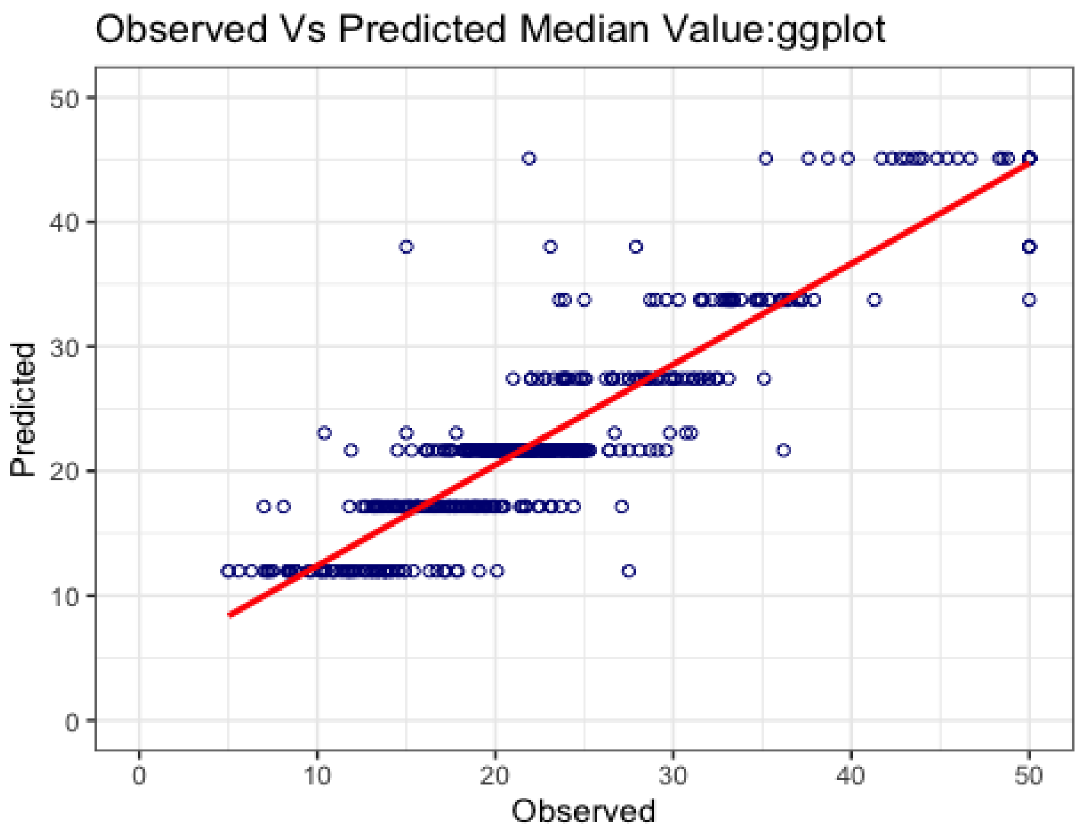
##Discussion:

- How many nodes does your tree have?
  - It has 9 nodes with 8 splits
- Did you prune the tree? Did it decrease the number of nodes?
  - yes, however, it didn’t reduce the number of nodes
- What is the prediction error (MSE)?
  - I have MSE of 13.31
- Plot the predicted vs. observed values. It appears that the variation between the predicted and actual increases as median value gets large
- Plot the final tree. The final tree has 9 nodes with 8 splits against our initial condition of minimum of 15 splits, though some nodes have observations number less than the specified minimum split which is concerning.

b) Apply bagging with 50 trees. Report the prediction error (MSE) and plot the predicted vs observed values.

```
## [1] 16.24467
```



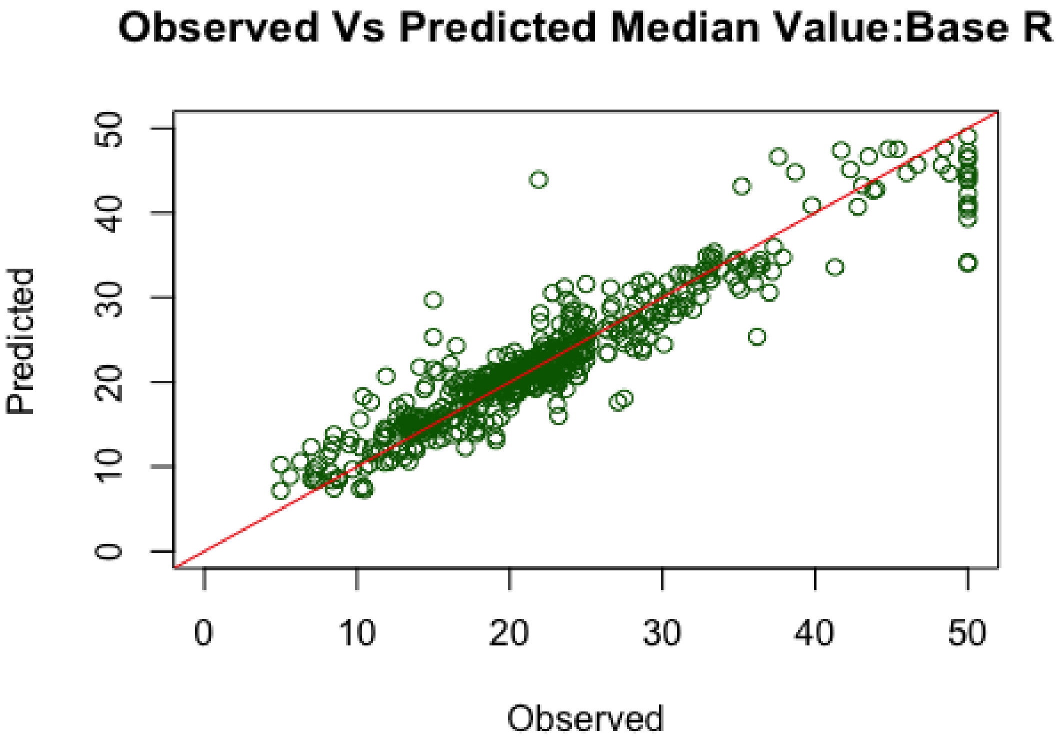


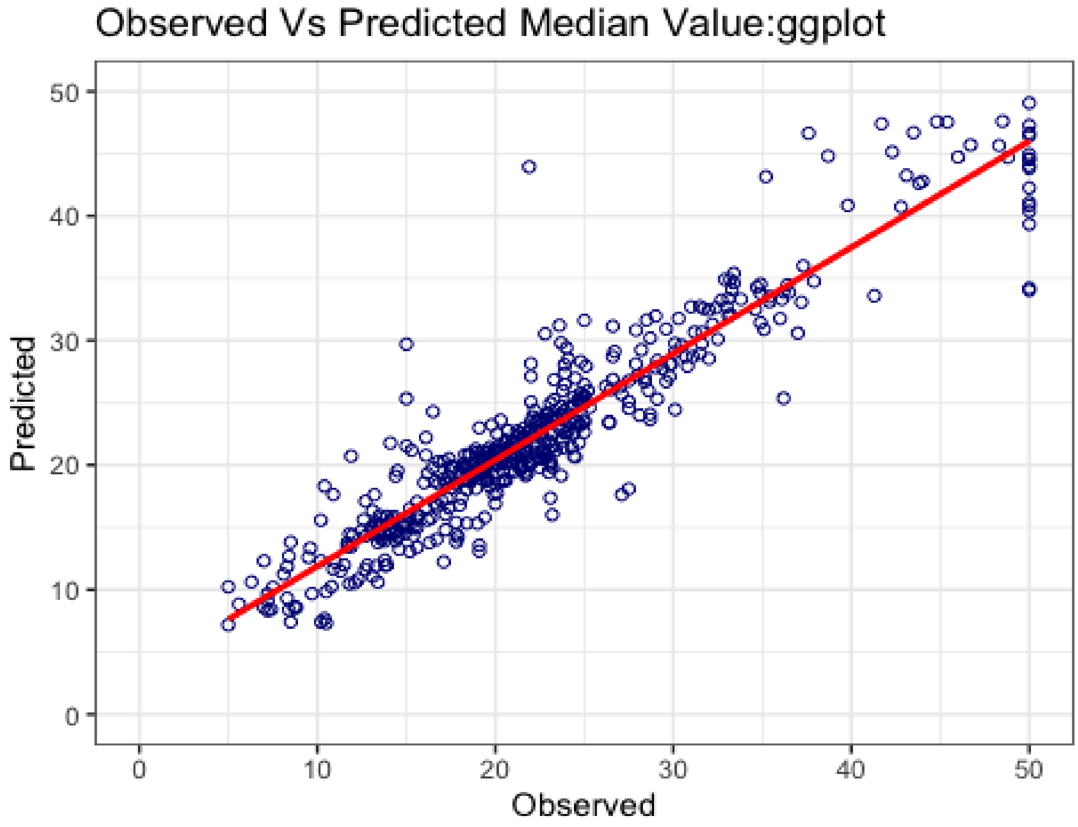
##Discussion:

I thought the bagging the decision tree would lead to a smaller mean square error because bagging is supposed to minimize the high variability,however, that isn't the case.We got a mean square error of 16.2 with the bagging method, perhaps bagging doesn't always improve a model.

c) Apply bagging using the randomForest() function. Report the prediction error (MSE). Was it the same as (b)? If they are different what do you think caused it? Plot the predicted vs. observed values.

```
## [1] 10.34007
```


Observed Vs Predicted Median Value:Base R
Predicted
0
10
20
30
40
50
0
10
20
30
40
50
Observed

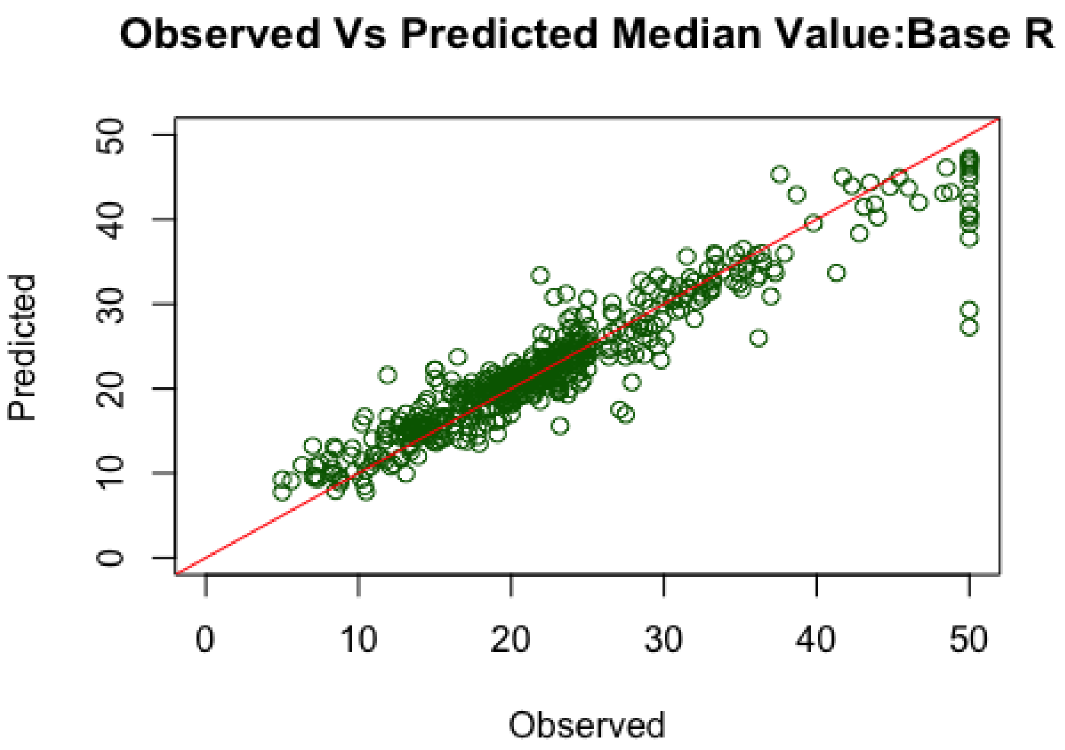

Observed Vs Predicted Median Value:ggplot
Predicted
0
10
20
30
40
50
0
10
20
30
40
50
Observed

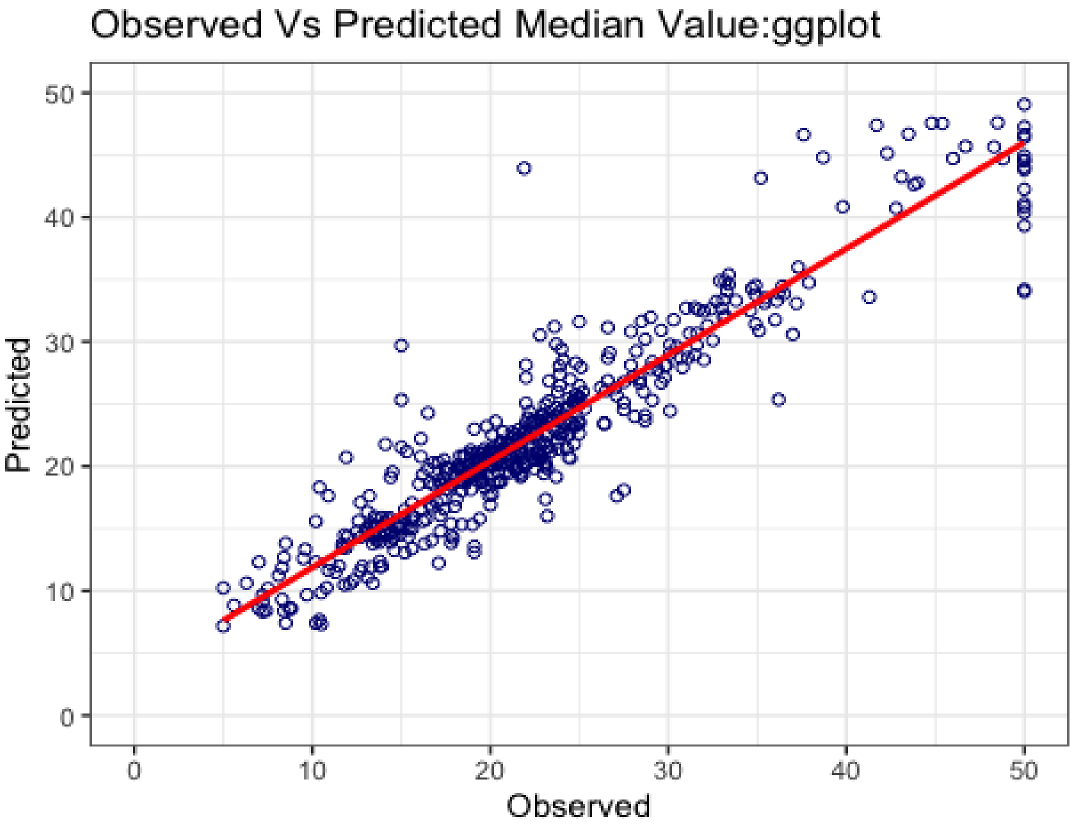
##Discussion:

During the random forest model construction I used 50 trees like I have in part b of the bagging method. Further more, I included all of the 13 candidate covariates instead of using the default method.Surprisingly, the mean square error is 10.34 which is smaller than both the rpart() regression tree in part a and the bagging in part b. The plot looks a lot more linear which makes sense for why our mse was low because random forest with bagging reduces the variability in the data by randomly selecting covariates to split.

d) Use the randomForest() function to perform random forest. Report the prediction error (MSE). Plot the predicted vs. observed values.

```
## [1] 9.750024
```





##Discussion:

I constructed this random forest model using the default mtry method which is the square root of the total number of the covariates. The plot looks more linear than the random forest with bagging method in part c. This indicates that random forest without bagging reduced the variability in the data even more which is why our mean square error (mse) is even lower. The mse of the random forest without bagging is 9.75 whereas the random forest with bagging model in part c had an mse of 10.34. Moving forward, I would suggest using random forest as it adds randomness in the training of the data especially during the explanatory variables selection.

e) Include a table of each method and associated MSE. Which method is more accurate?

```
##                   Method       MSE
## 1      Regression Tree 13.307879
## 2              Bagging 16.244674
## 3 Bagging Random Forest 10.340072
## 4        Random Forest  9.750024
```

##Discussion:

From the table it’s evident that random forest is the best alternative for predicting the median value of owner-occupied homes. It has an error rate of about 9.75% which is the lowest among the 4 machine learning models we built and tested. Having said that, bagging with random forest shouldn’t be discounted all together by simply comparing the mean square errors which isn’t that much worse than random forest. I believe different methods are more appropriate for different applications. However, in this particular Boston housing dataset, random forest gives the most reliable prediction with the lowest error.

##Works Cited

1. Michael, Semhar, and Christopher P. Saunders. “Recursive Partitioning.” Chapter 8. 10 Oct. 2020, South Dakota State University, South Dakota State University.
2. Therneau, Terry, and Elizabeth Atkinson.“An Introduction to Recursive Partitioning Using the RPART Routines”. 2015.
3. Boehmke, Bradley, and Brandon Greenwell. Chapter 10 Bagging | Hands-On Machine Learning with R. Bradleyboehmke.Github.Io, 1 Feb. 2020, bradleyboehmke.github.io/HOML/bagging.html. Accessed 14 Oct. 2020.
4. Steorts, Rebecca C. “Tree Based Methods: Bagging, Boosting, and Regression Trees.” STA 325, Chapter 8 ISL. 9 Oct. 2020, Duke University, Duke University.
5. Jackson, Simon. “Visualising Residuals • BlogR.” BlogR on Svbtle, drsimonj.svbtle.com/visualising-residuals.
6. Everitt, Brian, and Torsten Hothorn. A Handbook of Statistical Analyses Using n SECOND EDITION. Taylor and Francis Group LLC, 2010.