# Homework 5

Snigdha Peddi

**Question 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.

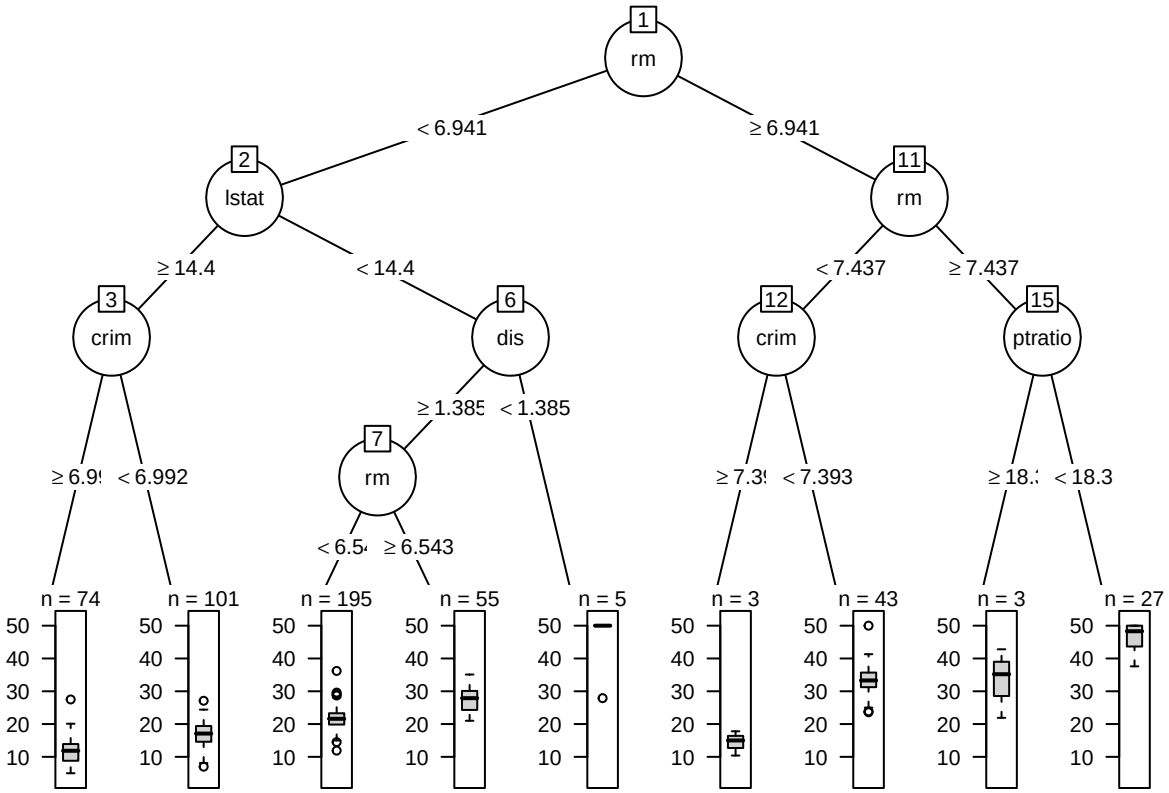
**a)** Construct a regression tree using rpart(). Discuss the results, including these key components:

- How many nodes does your tree have?
- Did you prune the tree? Did it decrease the number of nodes?
- What is the prediction error (MSE)?
- Plot the predicted vs. observed values.
- Plot the final tree.

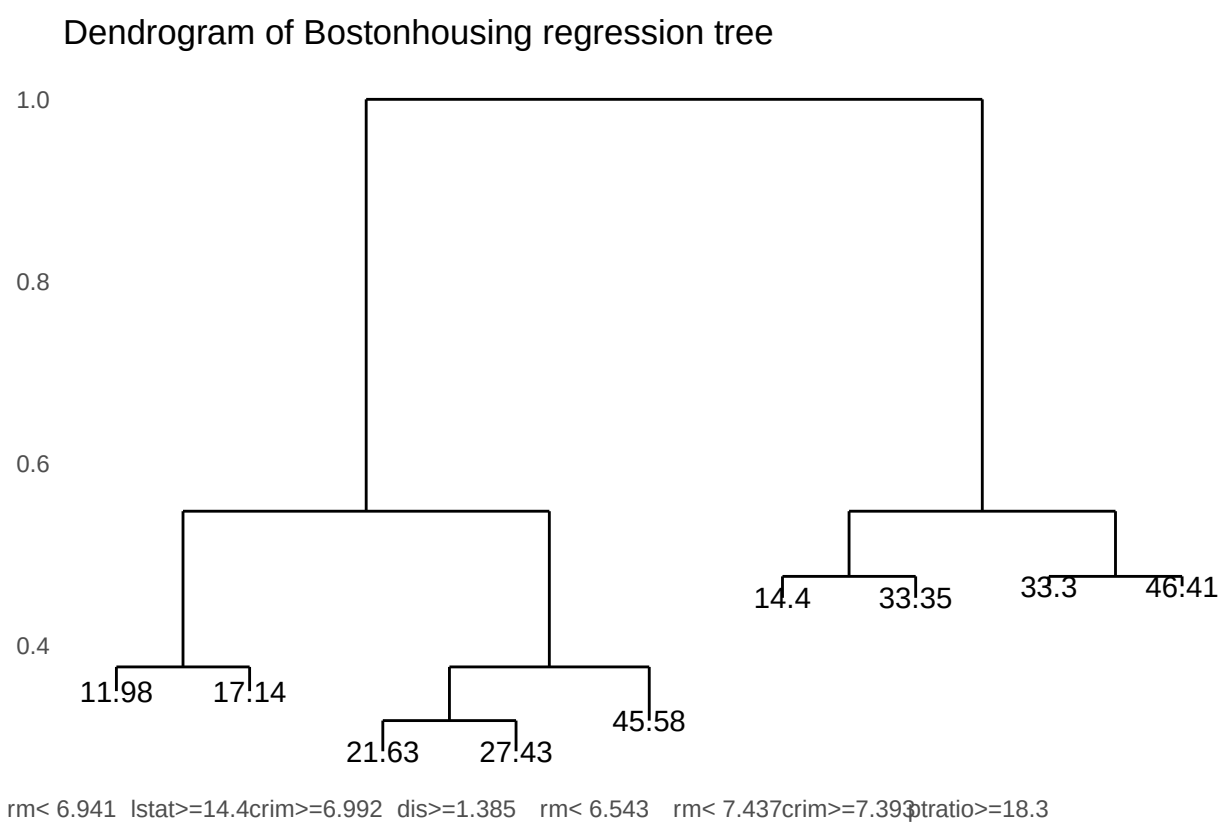
**References:** Ref1-Chapter_9Rcode.R, Ref2,Ref3, Ref4, Ref5,Ref6

**Answer 1.a:** Constructed a regression tree using rpart() function and plotted the tree using plot function of **partykit** package. Adjusted the font size to get a clear picture of the tree.

There is one Root node,7 Decision nodes and 9 Leaf nodes in the Tree.

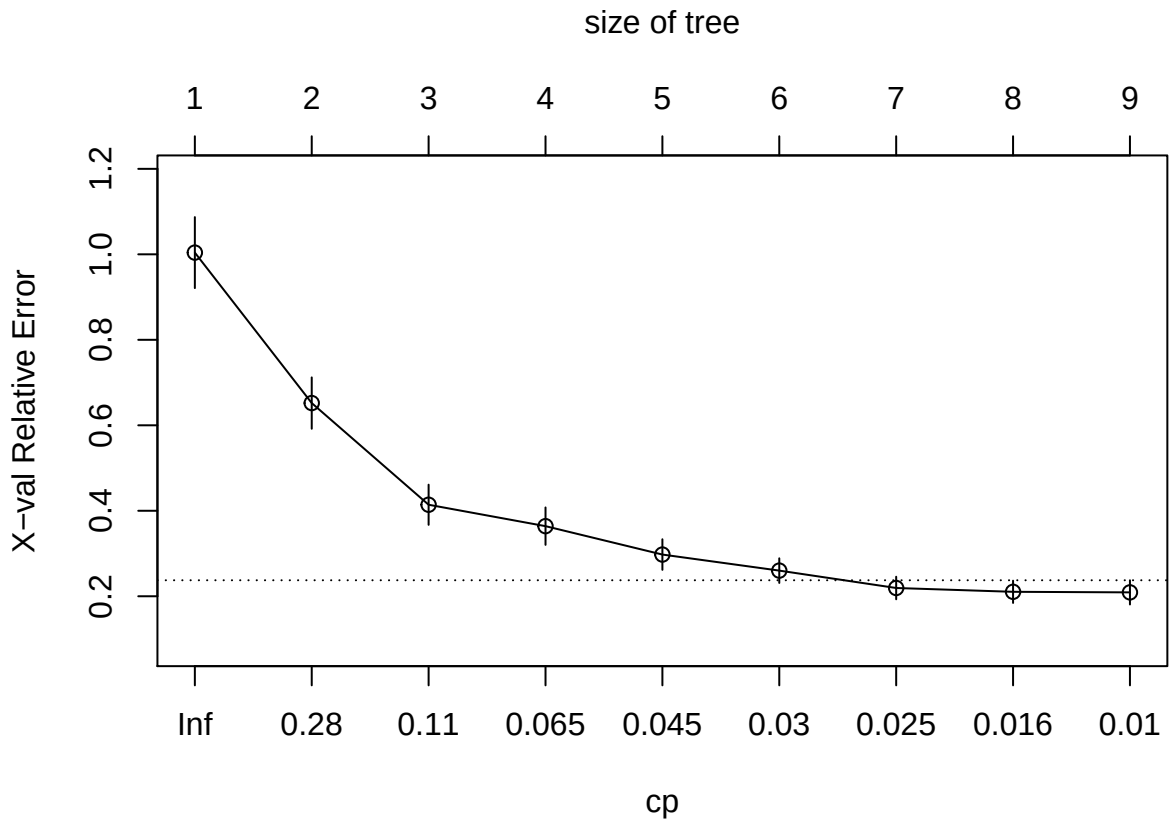



Used ggdendrogram function from **ggdendro** library to plot using ggplot2.



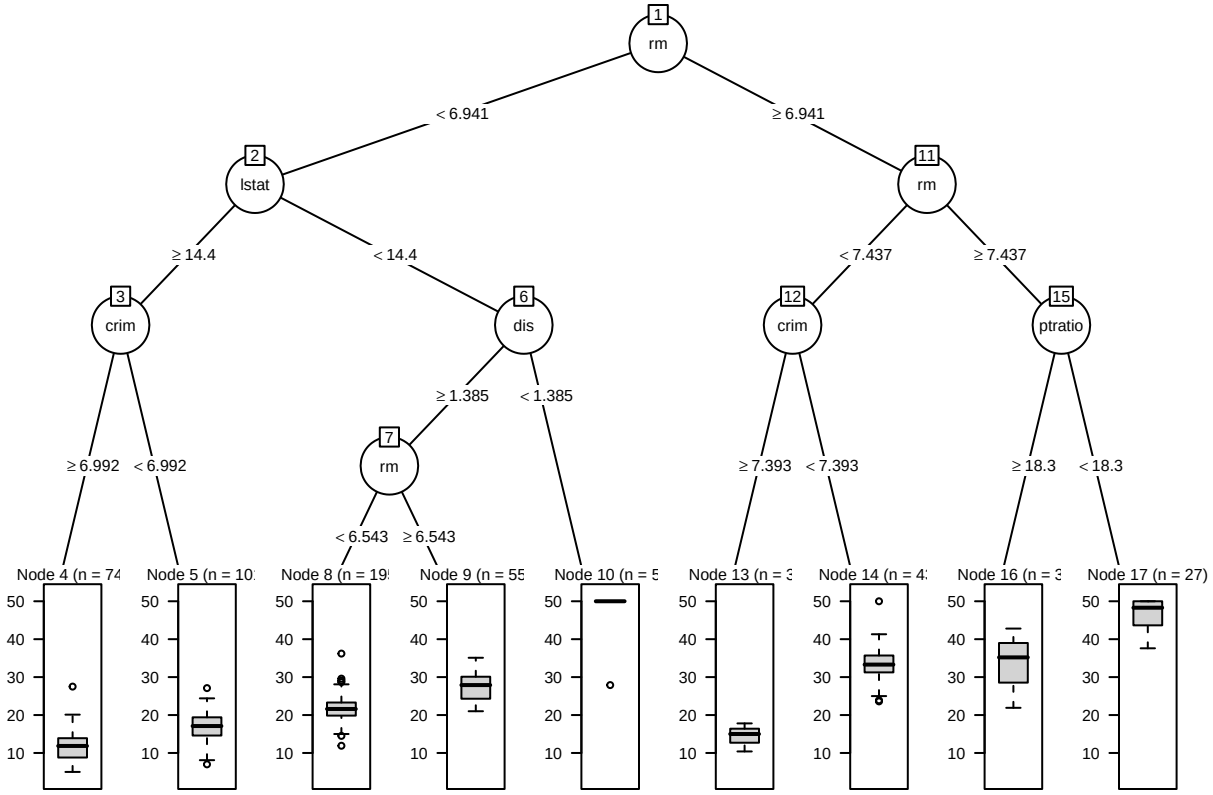

The Complexity Paramater and Cross validation error is decreasing consistently at each split indicating that no pruning is necessary.If we prune on basis of the lower cross validation error, then both the pruned tree and unpruned tree will look alike.

```
##
## Regression tree:
## rpart(formula = medv ~ ., data = Bostonhousing, control = rpart.control(minsplit = 10))
##
## 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 1.00405 0.083067
## 2 0.171172      1   0.54726 0.65200 0.060116
## 3 0.071658      2   0.37608 0.41410 0.047014
## 4 0.059002      3   0.30443 0.36401 0.043777
## 5 0.033756      4   0.24542 0.29759 0.035813
## 6 0.026613      5   0.21167 0.25999 0.028859
## 7 0.023572      6   0.18506 0.21933 0.026417
## 8 0.010859      7   0.16148 0.21040 0.026087
## 9 0.010000      8   0.15062 0.20908 0.028355
```



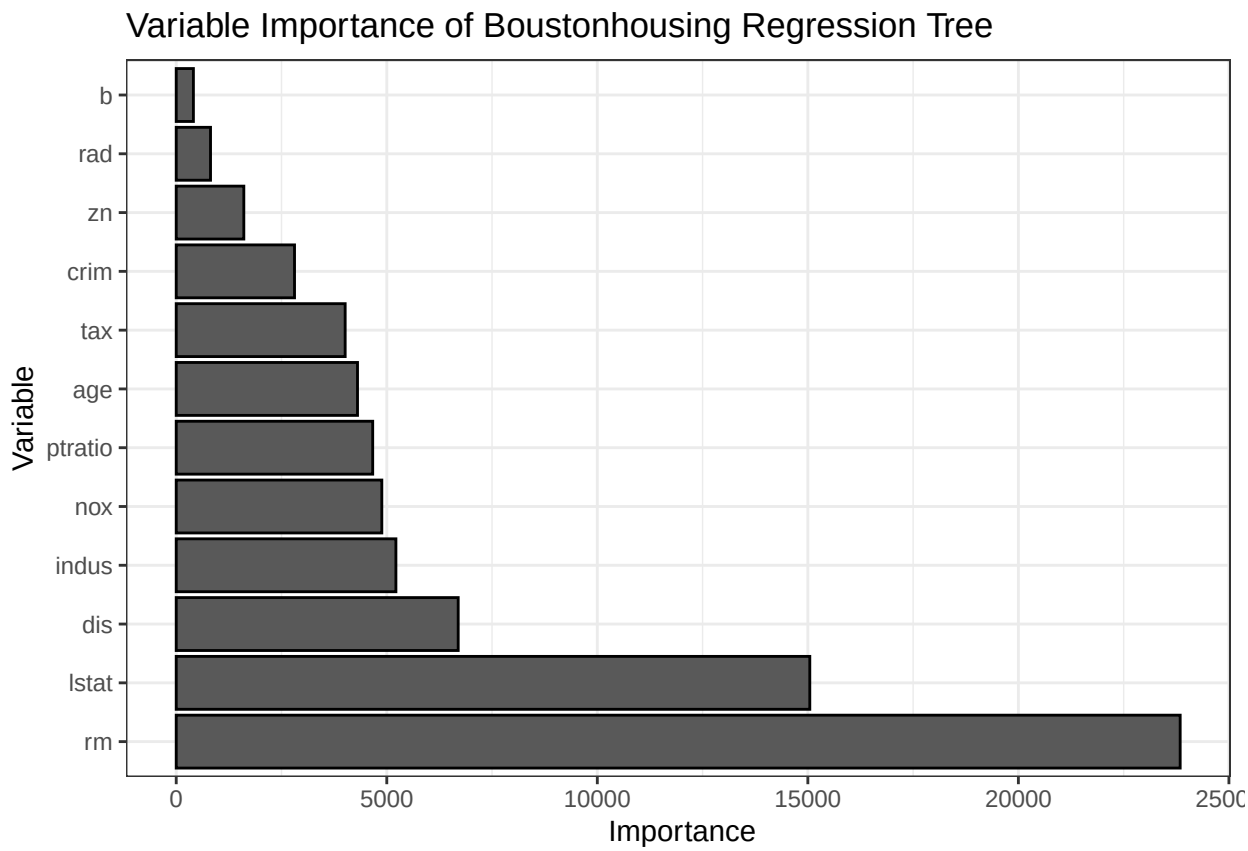

**Regression Tree after pruning based on minimum cross validation error**

```
##
## Regression tree:
## rpart(formula = medv ~ ., data = Bostonhousing, control = rpart.control(minsplit = 10))
##
## 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 1.00405 0.083067
## 2 0.171172      1   0.54726 0.65200 0.060116
## 3 0.071658      2   0.37608 0.41410 0.047014
## 4 0.059002      3   0.30443 0.36401 0.043777
## 5 0.033756      4   0.24542 0.29759 0.035813
## 6 0.026613      5   0.21167 0.25999 0.028859
## 7 0.023572      6   0.18506 0.21933 0.026417
## 8 0.010859      7   0.16148 0.21040 0.026087
## 9 0.010000      8   0.15062 0.20908 0.028355
```




The Variable Importance can be extracted using variable.importance parameter of fitted model to check which variables can be pruned to reduce the the cp and cross validation error.The variable importance is plotted using ggplot.

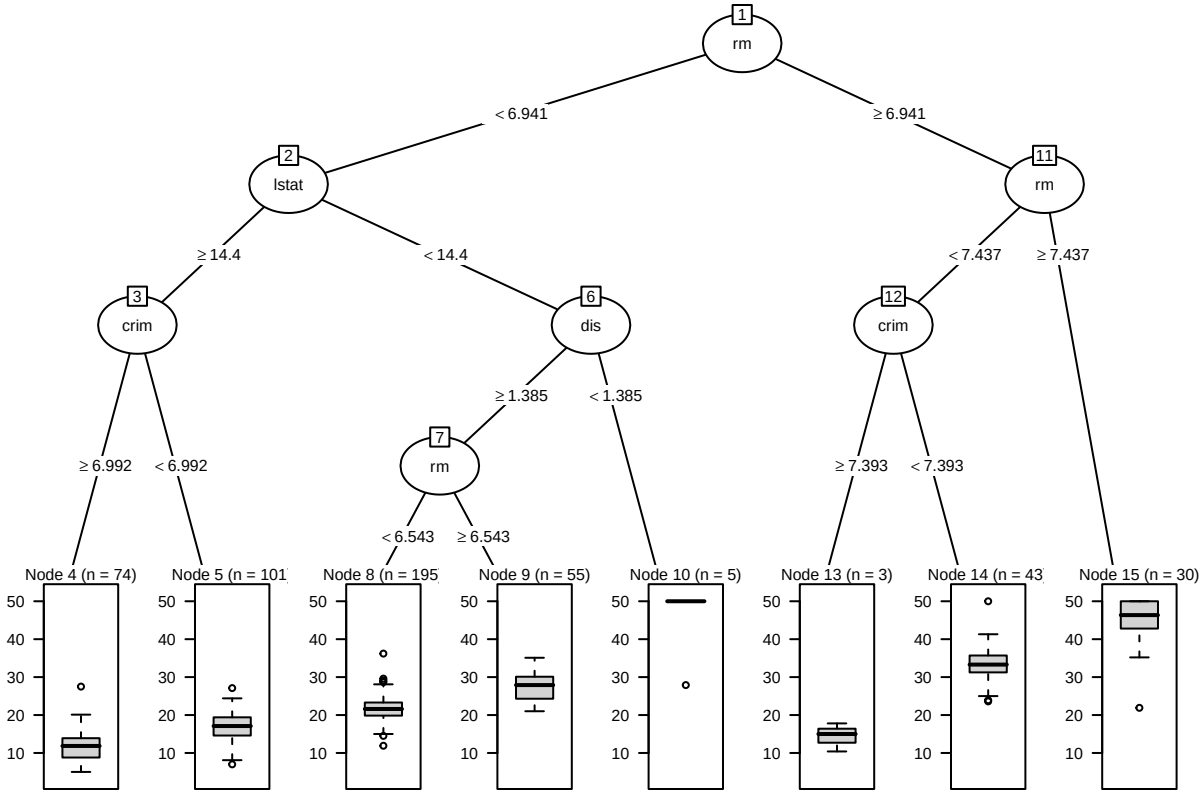
```
##         Bostonhousing_rpart$variable.importance
## rm                                    23842.4392
## lstat                                 15046.6276
## dis                                    6696.6716
## indus                                  5217.2458
## nox                                    4882.8608
## ptratio                                4666.1695
## age                                    4304.9329
## tax                                    4011.7488
## crim                                   2809.1645
## zn                                     1605.4575
## rad                                     814.2009
## b                                       408.1277
```




**Regression Tree after pruning based on minimum Standard Error**

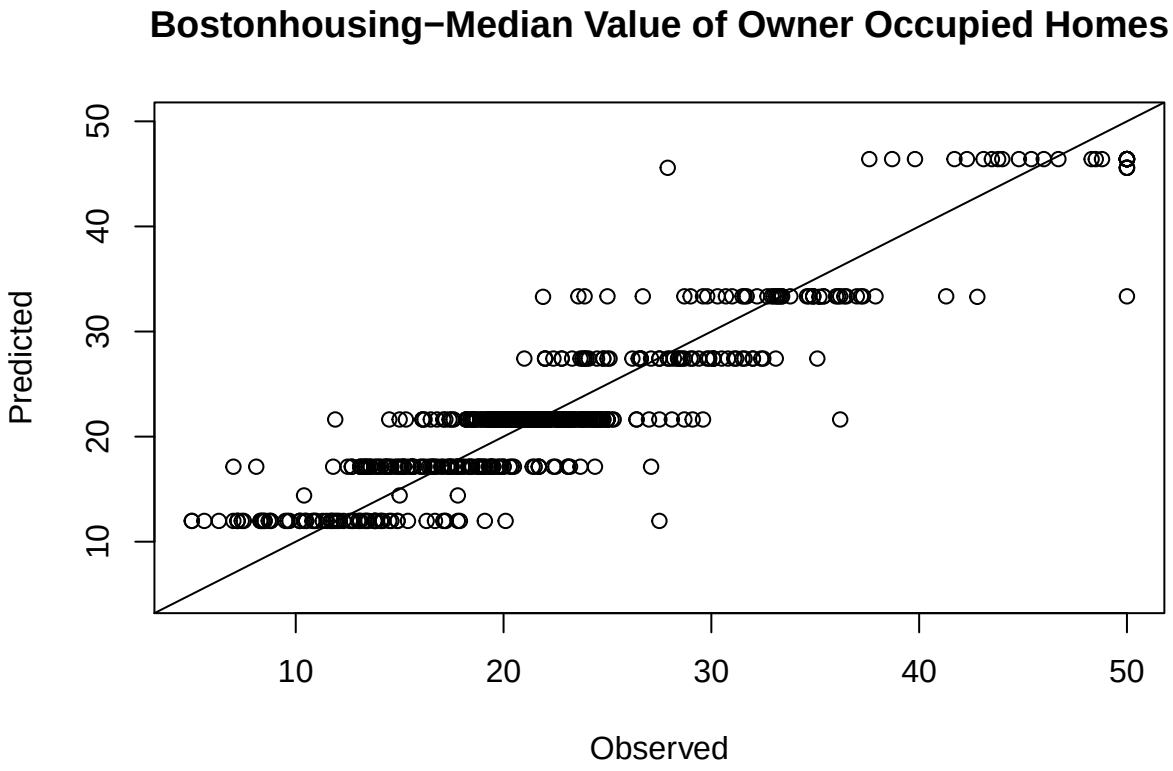
The standard error has increased after the 8th split and the tree can be pruned to lower the standard error by removing the last split based on ptratio variable.

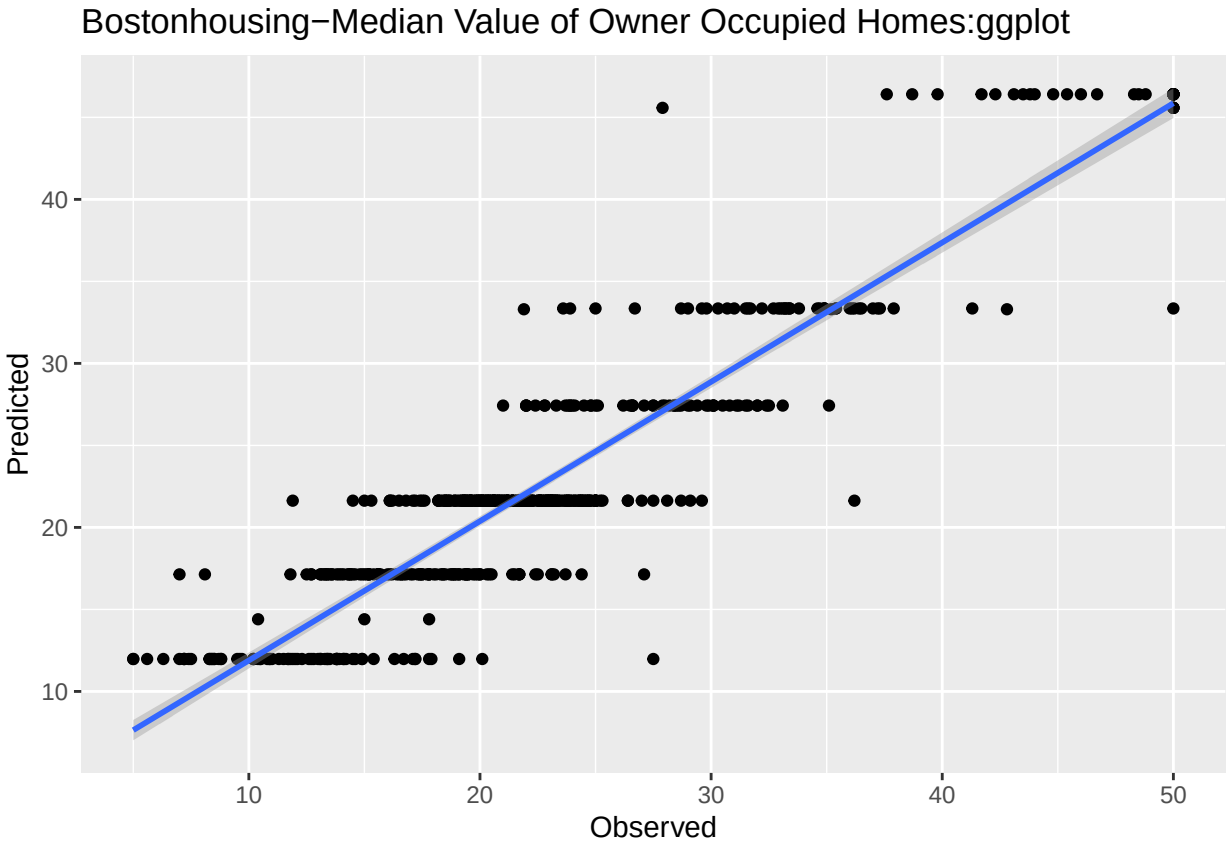
```
##
## Regression tree:
## rpart(formula = medv ~ ., data = Bostonhousing, control = rpart.control(minsplit = 10))
##
## Variables actually used in tree construction:
## [1] crim  dis   lstat rm
##
## Root node error: 42716/506 = 84.42
##
## n= 506
##
##         CP nsplit rel error  xerror     xstd
## 1 0.452744      0   1.00000 1.00405 0.083067
## 2 0.171172      1   0.54726 0.65200 0.060116
## 3 0.071658      2   0.37608 0.41410 0.047014
## 4 0.059002      3   0.30443 0.36401 0.043777
## 5 0.033756      4   0.24542 0.29759 0.035813
## 6 0.026613      5   0.21167 0.25999 0.028859
## 7 0.023572      6   0.18506 0.21933 0.026417
## 8 0.010859      7   0.16148 0.21040 0.026087
```




**Predicted vs Observed values**

Plotted the Predicted median values vs observed mean values.It is clear from the plot that there are few observations that are predicted incorrectly in each bin.


Bostonhousing−Median Value of Owner Occupied Homes
Predicted
Observed
10
20
30
40
50




**MSE**

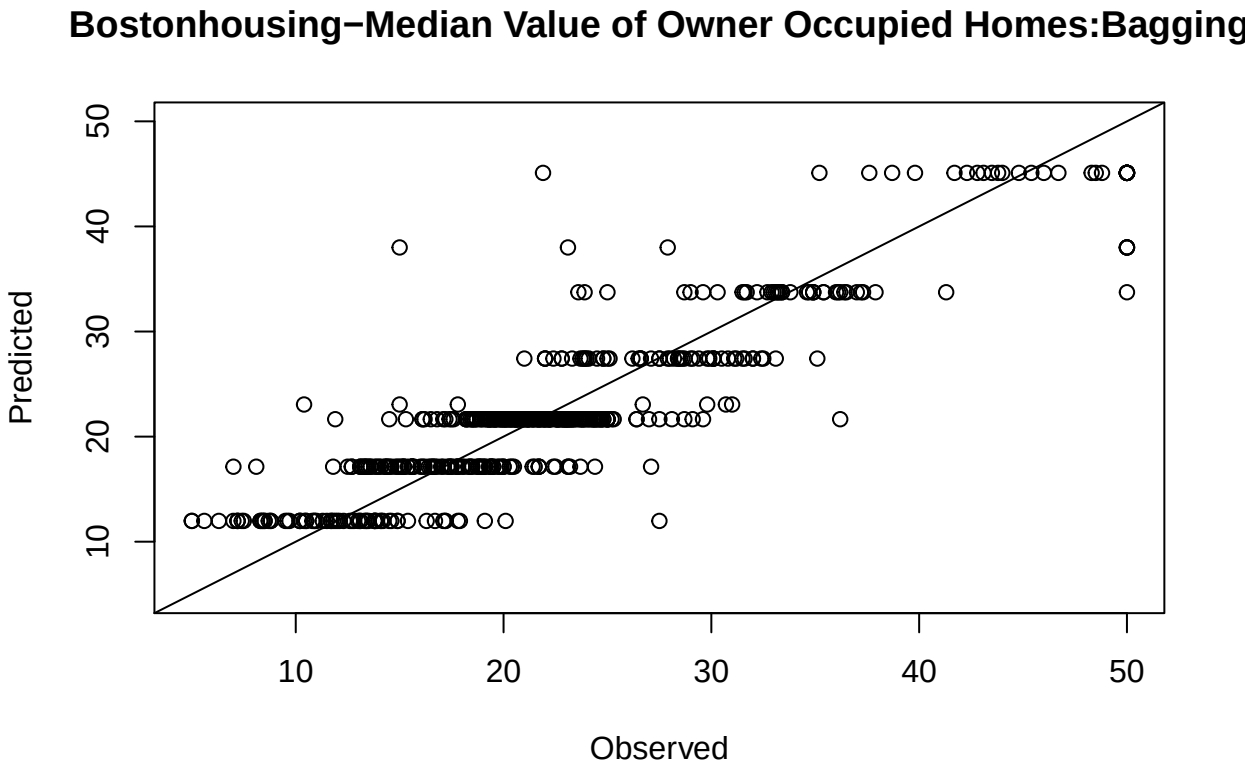
```
## MSE of the BostonHousing Regression Tree: 12.71556
```

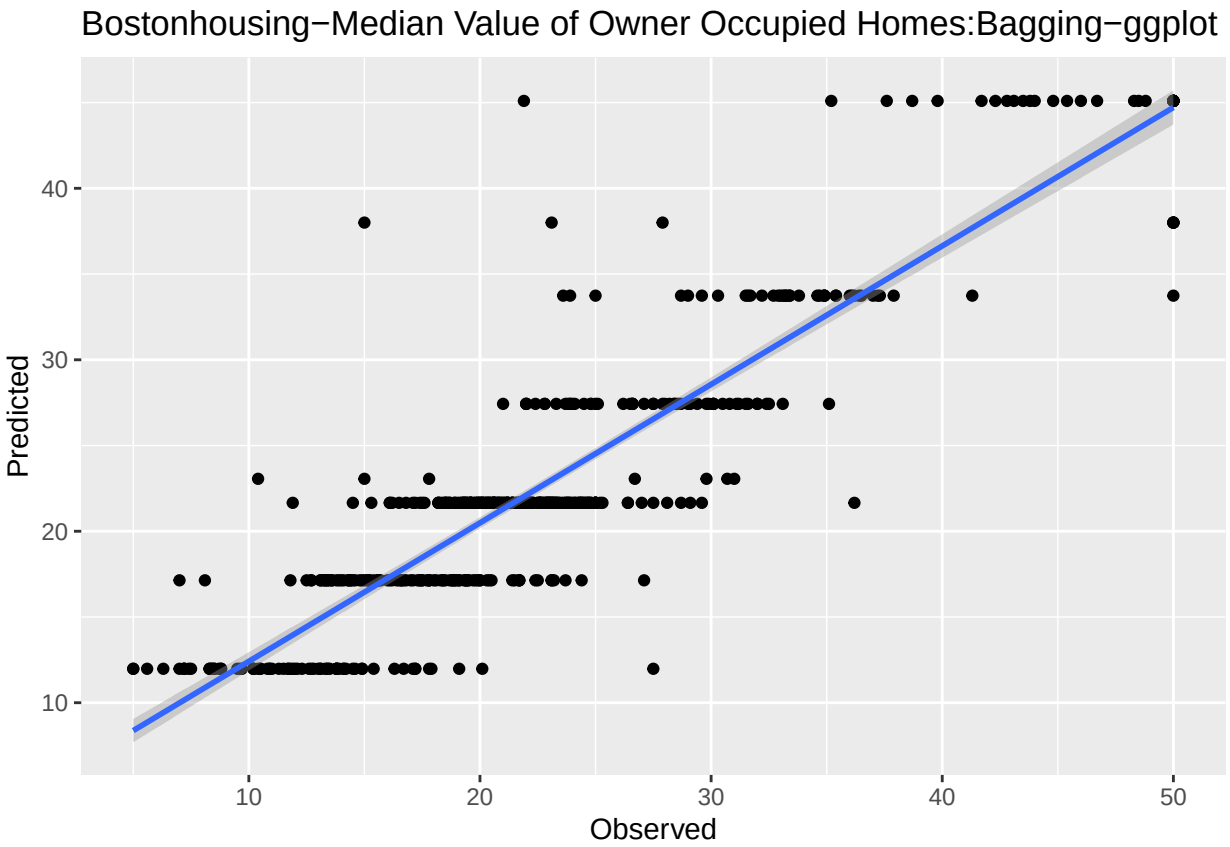
**Question 1.b:** Apply bagging with 50 trees. Report the prediction error (MSE) and plot the predicted vs observed values.

**References:** Ref1-Chapter_9Rcode.R

**Answer 1.b:** 50 base learners(decision trees) were modeled. A plot of Predicted median values and Observed values id generated.MSE of the results model is 16.24467.Though Boots trapping technique is used for bagging, all the features are considered for splitting a node of the base learners.This will results in the models which predict based on variables that are not correlated with the dependent variable and leading to high standard error.

```
## MSE of the BostonHousing model after Bagging with 50 trees: 16.24467
```


Bostonhousing−Median Value of Owner Occupied Homes:Bagging
Predicted
Observed
10
20
30
40
50

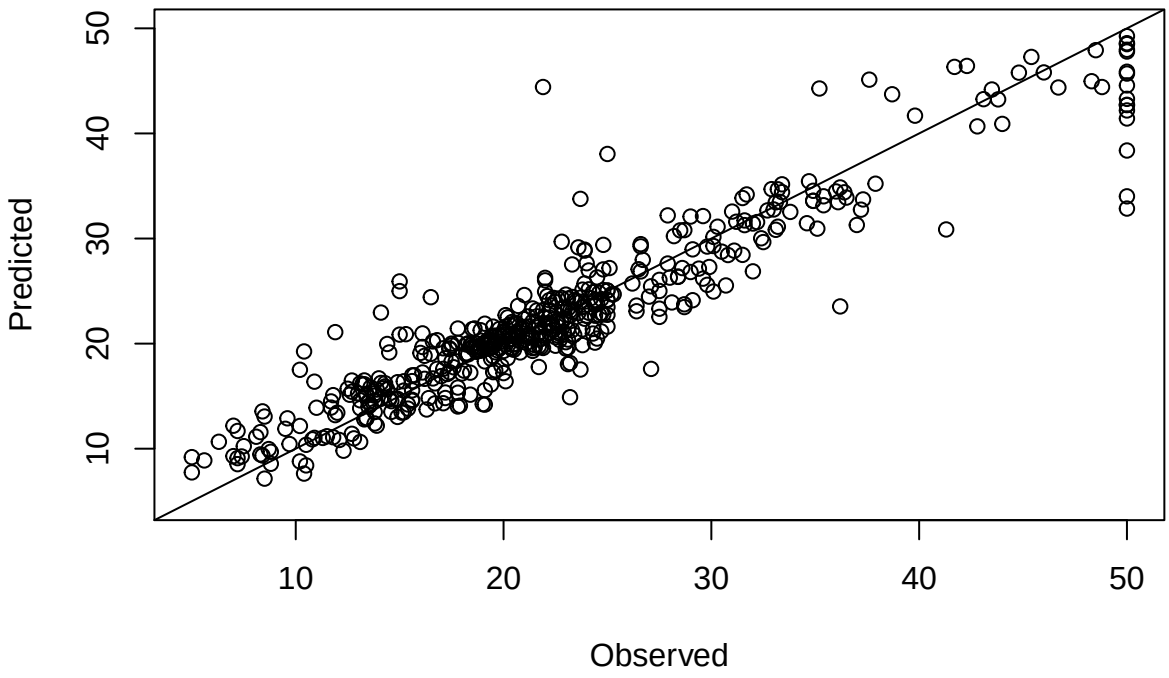


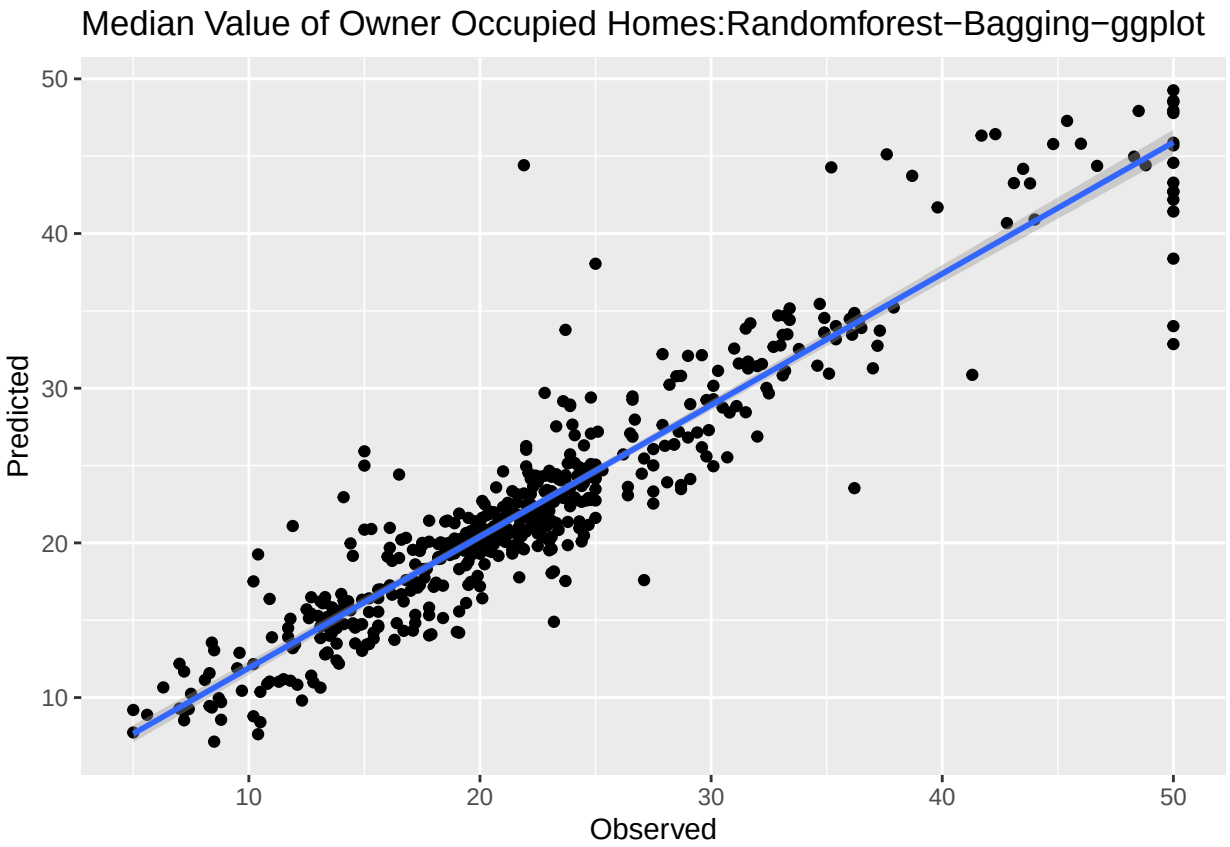

**Question 1.c:** Apply bagging using the randomForest() function. Report the prediction error (MSE). Was it the same as (b)? If they are different what do you think caused it? Plot the predicted vs. observed values.

**References:** Ref1-Chapter_9Rcode.R,Ref6,Ref7

**Answer 1.C:** In randomforest the data for the base learners is extracted by bootstrapping (row sampling with replacement technique) and predictions were made by aggregation.Only part of the training data is used by each learner.This will reduce the overfitting and overall high variance caused by individual base learner(decision tree in this case). The MSE of the bagging model using randomForest function is 10.6458 and is lower than the bagging using rpart() function.This is because the randomForest function uses best split feature to split each node of the tree rather than considering all the features thus reducing the overall variance. The plot between predicted and observed median values of home owners shows that majority of predictions are close to the real values though there are outliers.

```
## MSE of the BostonHousing Regression Tree: 10.6458
```


Median Value of Owner Occupied Homes:Randomforest−Bagging
Predicted
Observed
10
20
30
40
50

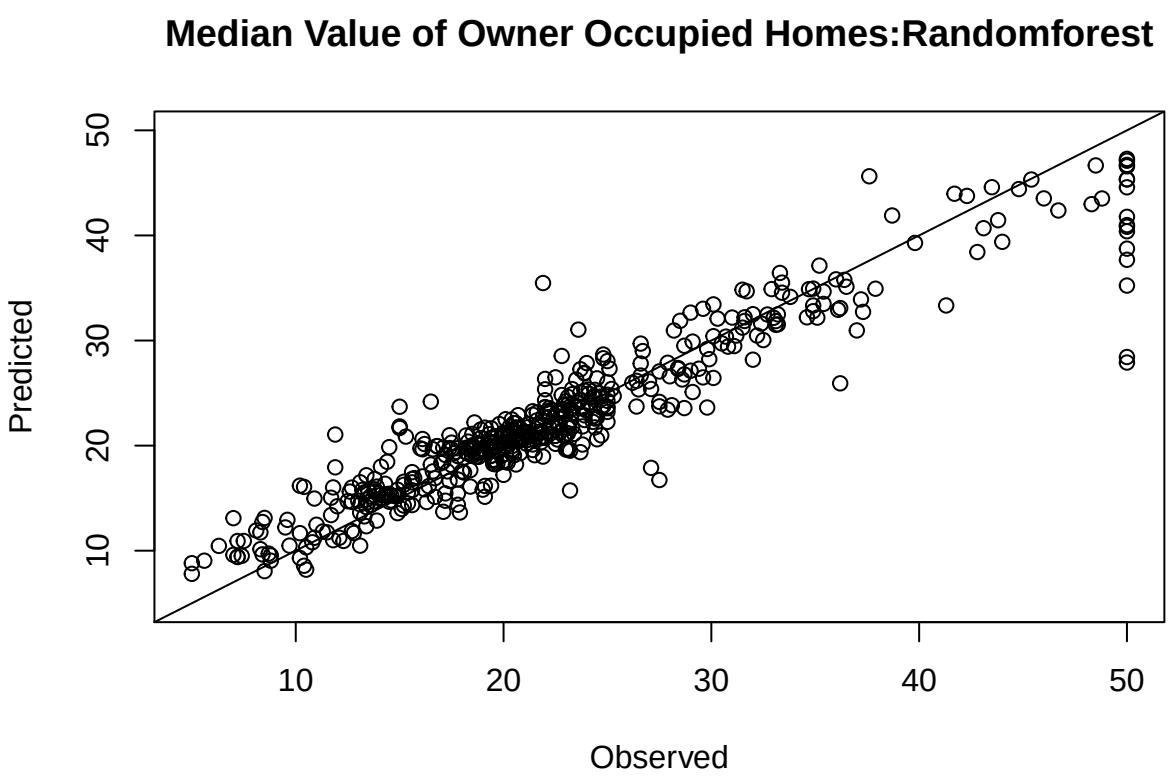


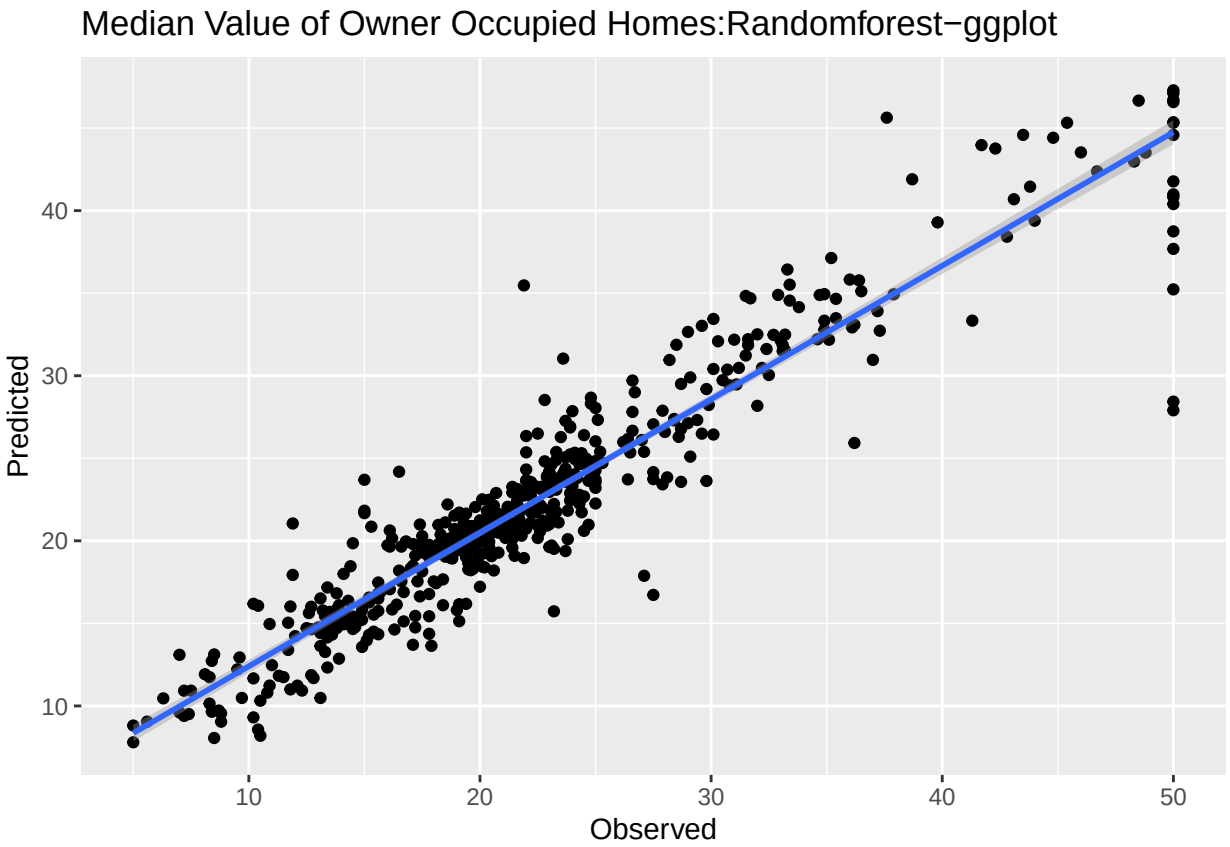

**Question 1.d:** Use the randomForest() function to perform random forest. Report the prediction error (MSE). Plot the predicted vs. observed values.

**References:** Ref1-Chapter_9Rcode.R,Ref7

**Answer 1.d:** The lower MSE of the model using randomForest function than the MSE using random forest and bagging is due to the Feature sampling at each node of the tree. Only few features are randomly selected in each base learner.

```
## MSE of the BostonHousing Regression Tree: 10.09328
```


Median Value of Owner Occupied Homes:Randomforest
Predicted
Observed
10
20
30
40
50




**Question 1.e)** Include a table of each method and associated MSE. Which method is more accurate?

**Answer 1.e:** The comparison of MSE of all the methods shows that Random forest is more accurate than other methods.This is due to bootstrapping aggregation along with feature sampling at each node.As random sample of data is used by each model and best variable is used to split the tree at each node, it returns better predictions and reduce overfitting.

```
##   RandomForest RandomForest_bagging  Bagging    rpart
## 1     10.09328              10.6458 16.24467 12.71556
```