Random Forest Exercise - Kinjal Majumdar

```
suppressMessages(library(ipred))
```

```
## Warning: package 'ipred' was built under R version 3.5.2
```

```
suppressMessages(library(randomForest))
```

```
## Warning: package 'randomForest' was built under R version 3.5.2
```

```
suppressMessages(library(tidyverse))
```

```
## Warning: package 'tidyverse' was built under R version 3.5.2
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Warning: package 'tibble' was built under R version 3.5.2
```

```
## Warning: package 'tidyr' was built under R version 3.5.2
```

```
## Warning: package 'readr' was built under R version 3.5.2
```

```
## Warning: package 'purrr' was built under R version 3.5.2
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
## Warning: package 'forcats' was built under R version 3.5.2
```

```
suppressMessages(library(ggplot2))
suppressMessages(library(tree))
```

```
## Warning: package 'tree' was built under R version 3.5.2
```

```
suppressMessages(library(ISLR))
```

```
## Warning: package 'ISLR' was built under R version 3.5.2
```

```
suppressMessages(library(adabag))
```

```
## Warning: package 'adabag' was built under R version 3.5.2
```

```
## Warning: package 'rpart' was built under R version 3.5.2
```

```
## Warning: package 'caret' was built under R version 3.5.2
```

```
## Warning: package 'foreach' was built under R version 3.5.2
```

```
## Warning: package 'doParallel' was built under R version 3.5.2
```

```
## Warning: package 'iterators' was built under R version 3.5.2
```

```
suppressMessages(library(rpart))
suppressMessages(attach(Carseats))
str(Carseats)
```

```
## 'data.frame':    400 obs. of  11 variables:
##  $ Sales      : num  9.5 11.22 10.06 7.4 4.15 ...
##  $ CompPrice  : num  138 111 113 117 141 124 115 136 132 132 ...
##  $ Income     : num  73 48 35 100 64 113 105 81 110 113 ...
##  $ Advertising: num  11 16 10 4 3 13 0 15 0 0 ...
##  $ Population : num  276 260 269 466 340 501 45 425 108 131 ...
##  $ Price      : num  120 83 80 97 128 72 108 120 124 124 ...
##  $ ShelveLoc  : Factor w/ 3 levels "Bad","Good","Medium": 1 2 3 3 1 1 3 2 3 3 ...
##  $ Age        : num  42 65 59 55 38 78 71 67 76 76 ...
##  $ Education  : num  17 10 12 14 13 16 15 10 10 17 ...
##  $ Urban      : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 1 ...
##  $ US         : Factor w/ 2 levels "No","Yes": 2 2 2 2 1 2 1 2 1 2 ...
```

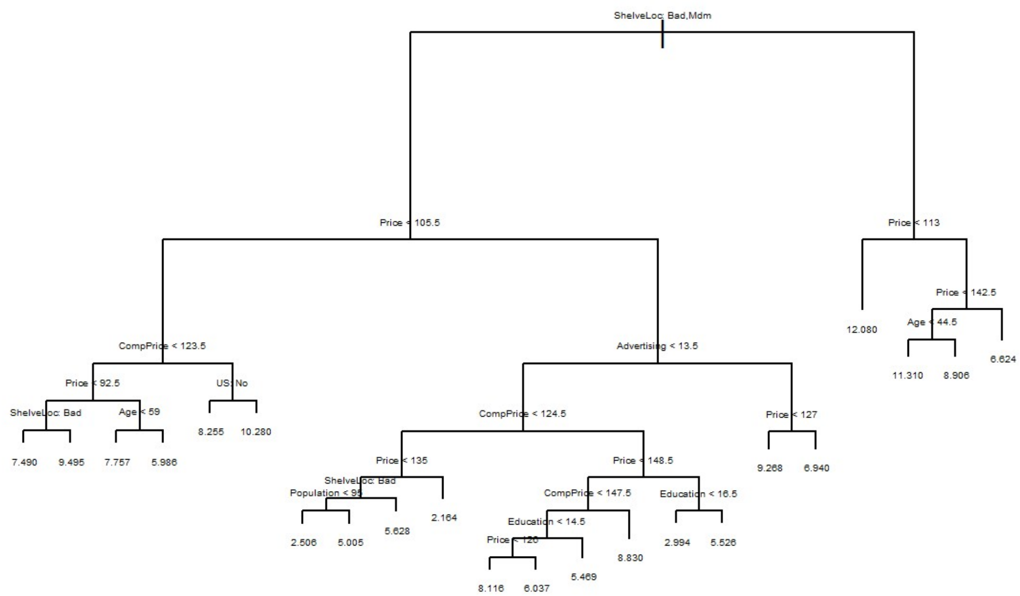
```r
#Tree for one variable
tree.seats = tree(Sales ~ ., data = Seats.train)

#Check summary and structure
summary(tree.seats)
```

```
##
## Regression tree:
## tree(formula = Sales ~ ., data = Seats.train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "CompPrice"   "Age"         "US"
## [6] "Advertising" "Population"  "Education"
## Number of terminal nodes:  22
## Residual mean deviance:  2.197 = 391.1 / 178
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -4.60000 -0.81100 -0.08162  0.00000  0.72540  4.35300
```

```r
str(tree.seats)
```

```
## List of 6
##  $ frame  :'data.frame': 43 obs. of  5 variables:
##   ..$ var   : Factor w/ 11 levels "<leaf>","CompPrice",..: 7 6 2 6 7 1 1 8 1 1 ...
##   ..$ n     : num [1:43] 200 158 59 41 18 7 11 23 9 14 ...
##   ..$ dev   : num [1:43] 1559.2 1007.2 254.2 162.3 56.5 ...
##   ..$ yval  : num [1:43] 7.45 6.83 8.19 7.57 8.71 ...
##   ..$ splits: chr [1:43, 1:2] ":ac" "<105.5" "<123.5" "<92.5" ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : NULL
##   .. .. ..$ : chr [1:2] "cutleft" "cutright"
##  $ where  : Named int [1:200] 6 7 41 42 30 36 6 42 27 21 ...
##   ..- attr(*, "names")= chr [1:200] "81" "274" "365" "113" ...
##  $ terms  :Classes 'terms', 'formula'  language Sales ~ CompPrice + Income + Advert
ising + Population + Price + ShelveLoc +      Age + Education + Urban + US
##   .. ..- attr(*, "variables")= language list(Sales, CompPrice, Income, Advertising,
Population, Price, ShelveLoc,      Age, Education, Urban, US)
##   .. ..- attr(*, "factors")= int [1:11, 1:10] 0 1 0 0 0 0 0 0 0 0 ...
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : chr [1:11] "Sales" "CompPrice" "Income" "Advertising" ...
##   .. .. .. ..$ : chr [1:10] "CompPrice" "Income" "Advertising" "Population" ...
##   .. ..- attr(*, "term.labels")= chr [1:10] "CompPrice" "Income" "Advertising" "Pop
ulation" ...
##   .. ..- attr(*, "order")= int [1:10] 1 1 1 1 1 1 1 1 1 1
##   .. ..- attr(*, "intercept")= int 1
##   .. ..- attr(*, "response")= int 1
##   .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. ..- attr(*, "predvars")= language list(Sales, CompPrice, Income, Advertising,
Population, Price, ShelveLoc,      Age, Education, Urban, US)
##   .. ..- attr(*, "dataClasses")= Named chr [1:11] "numeric" "numeric" "numeric" "nu
meric" ...
##   .. .. ..- attr(*, "names")= chr [1:11] "Sales" "CompPrice" "Income" "Advertising"
...
##  $ call   : language tree(formula = Sales ~ ., data = Seats.train)
##  $ y      : Named num [1:200] 8.01 10.04 10.5 6.67 7.96 ...
##   ..- attr(*, "names")= chr [1:200] "81" "274" "365" "113" ...
##  $ weights: num [1:200] 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "class")= chr "tree"
##  - attr(*, "xlevels")=List of 10
##   ..$ CompPrice  : NULL
##   ..$ Income     : NULL
##   ..$ Advertising: NULL
##   ..$ Population : NULL
##   ..$ Price      : NULL
##   ..$ ShelveLoc  : chr [1:3] "Bad" "Good" "Medium"
##   ..$ Age        : NULL
##   ..$ Education  : NULL
##   ..$ Urban      : chr [1:2] "No" "Yes"
##   ..$ US         : chr [1:2] "No" "Yes"
```

```
#Create a plot of the tree model
plot(tree.seats)
text(tree.seats, pretty = 3, cex=0.3)
```



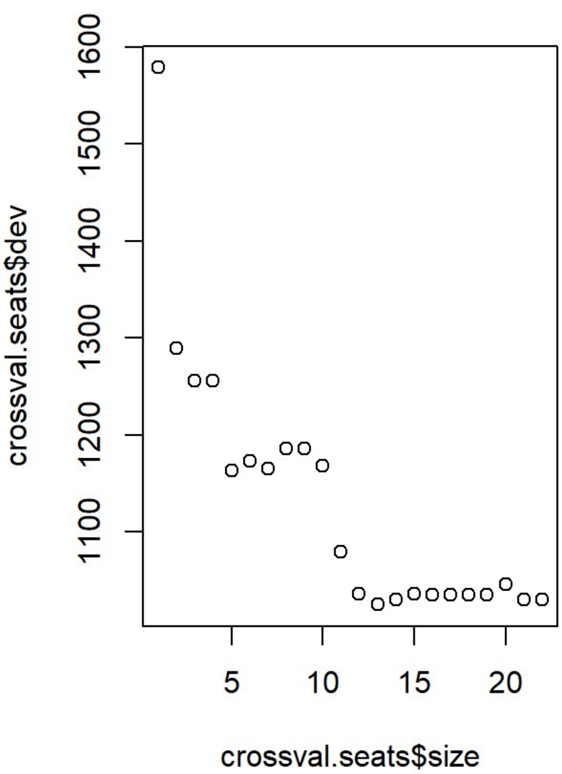

```
print(tree.seats)
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
##   1) root 200 1559.000  7.454
##     2) ShelveLoc: Bad,Medium 158 1007.000  6.832
##       4) Price < 105.5 59  254.200  8.194
##         8) CompPrice < 123.5 41  162.300  7.573
##          16) Price < 92.5 18   56.450  8.715
##            32) ShelveLoc: Bad 7    2.874  7.490 *
##            33) ShelveLoc: Medium 11   36.390  9.495 *
##          17) Price > 92.5 23   64.040  6.679
##            34) Age < 59 9   27.710  7.757 *
##            35) Age > 59 14   19.160  5.986 *
##         9) CompPrice > 123.5 18   40.130  9.607
##          18) US: No 6    2.568  8.255 *
##          19) US: Yes 12   21.100 10.280 *
##       5) Price > 105.5 99  578.400  6.021
##        10) Advertising < 13.5 81  419.700  5.558
##          20) CompPrice < 124.5 29  118.300  4.363
##            40) Price < 135 24   76.820  4.822
##              80) ShelveLoc: Bad 11   25.110  3.869
##               160) Population < 95 5    5.997  2.506 *
##               161) Population > 95 6    2.077  5.005 *
##              81) ShelveLoc: Medium 13   33.290  5.628 *
##            41) Price > 135 5   12.260  2.164 *
##          21) CompPrice > 124.5 52  237.000  6.223
##            42) Price < 148.5 42  130.200  6.691
##              84) CompPrice < 147.5 35   86.900  6.263
##               168) Education < 14.5 16   40.610  7.206
##                 336) Price < 126 9   11.490  8.116 *
##                 337) Price > 126 7   12.110  6.037 *
##               169) Education > 14.5 19   20.080  5.469 *
##              85) CompPrice > 147.5 7    4.867  8.830 *
##            43) Price > 148.5 10   59.070  4.260
##              86) Education < 16.5 5   34.520  2.994 *
##              87) Education > 16.5 5    8.526  5.526 *
##        11) Advertising > 13.5 18   63.230  8.104
##          22) Price < 127 9    4.139  9.268 *
##          23) Price > 127 9   34.710  6.940 *
##     3) ShelveLoc: Good 42  260.600  9.795
##       6) Price < 113 13   32.190 12.080 *
##       7) Price > 113 29  129.800  8.770
##        14) Price < 142.5 22   65.330  9.452
##          28) Age < 44.5 5    3.999 11.310 *
##          29) Age > 44.5 17   39.050  8.906 *
##        15) Price > 142.5 7   22.010  6.624 *
```

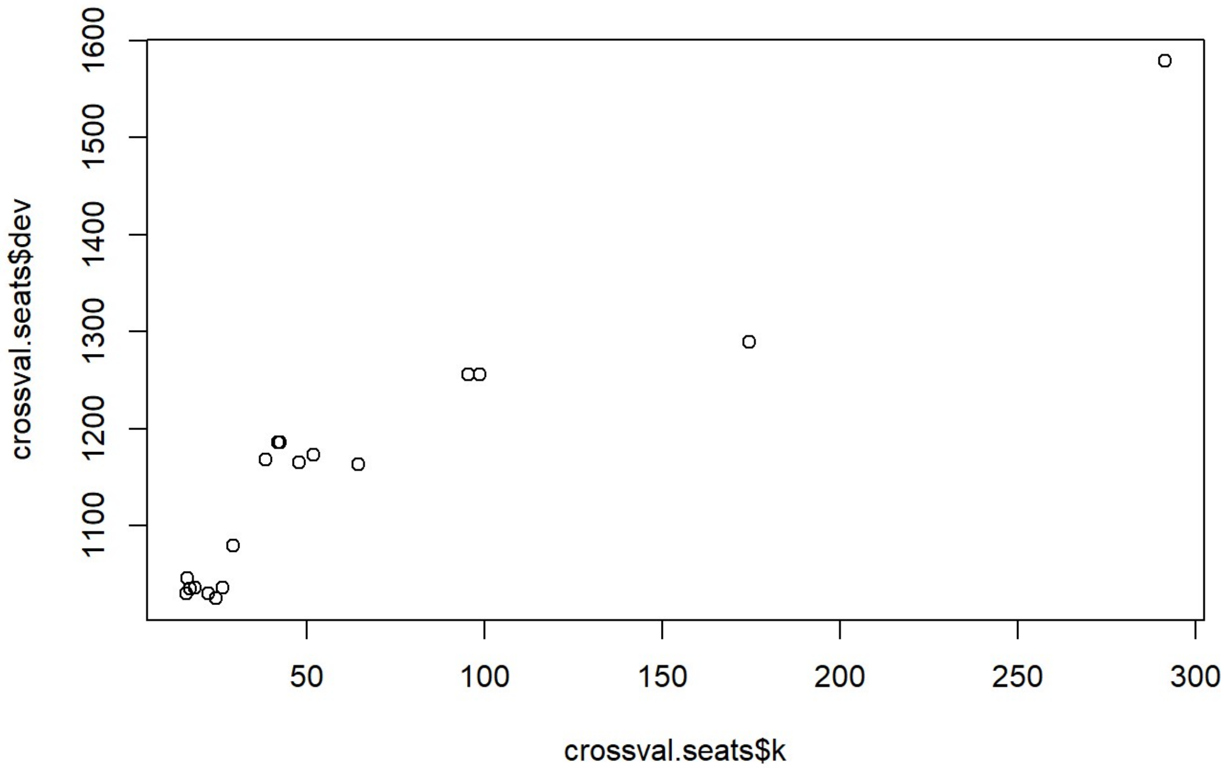
```
#Calculate Mean Squared Error
predict <- predict(tree.seats, Seats.test)
MSE <- mean((Seats.test$Sales-predict)^2)
MSE
```

```
## [1] 4.10117
```

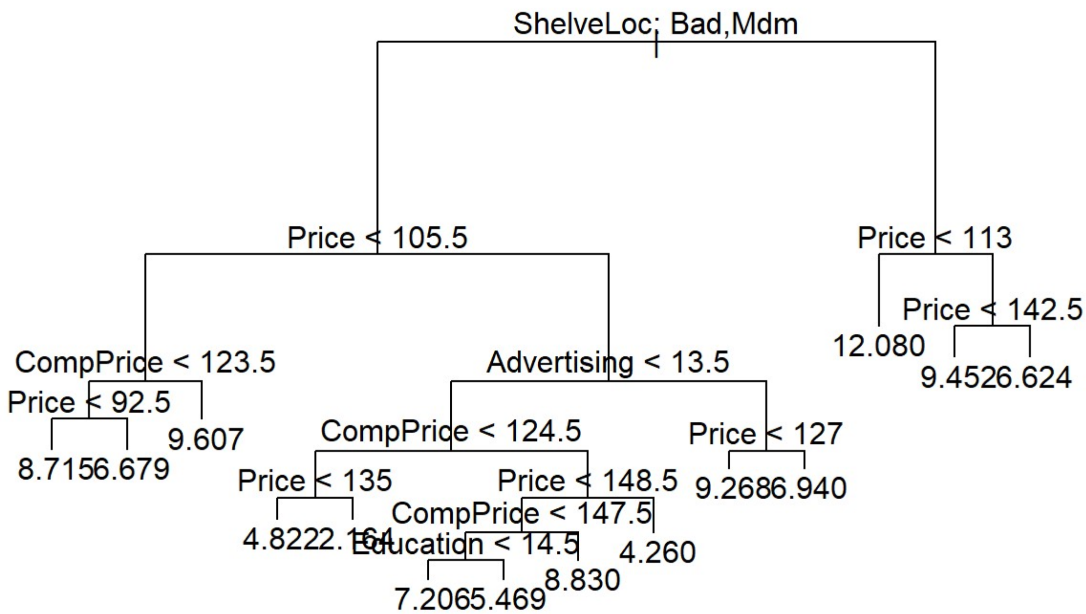
```
#Carry out Cross Validation
crossval.seats = cv.tree(tree.seats, FUN = prune.tree)
par(mfrow = c(1, 2))

plot(crossval.seats$size, crossval.seats$dev)
```




```
plot(crossval.seats$k, crossval.seats$dev)
```




```
# The optimal number as seen from the plot above is at 14
#Prune the tree
pruned.seats = prune.tree(tree.seats, best = 14)
plot(pruned.seats)
text(pruned.seats, pretty = 3)
```




```
pr.predict <- predict(pruned.seats, Seats.test)
MSE.pr <- mean((Seats.test$Sales-pr.predict)^2)
print(MSE.pr)
```

```
## [1] 4.609877
```

```
#Carry out bagging
bag.seats = randomForest(Sales ~ ., data = Seats.train, mtry = 8, ntree = 310,
    importance = T)
predict.bag = predict(bag.seats, Seats.test)
MSE.bag<- mean((Seats.test$Sales - predict.bag)^2)
MSE.bag
```

```
## [1] 2.498624
```

```
print(importance(bag.seats))
```

```
##               %IncMSE IncNodePurity
## CompPrice   14.584235    159.651388
## Income       2.818852     75.306165
## Advertising 10.371472    130.245109
## Population   1.819496     92.864383
## Price       40.552484    494.684815
## ShelveLoc   37.340574    331.098577
## Age          8.919421    138.144386
## Education    2.805197     54.862179
## Urban       -1.031310      7.639277
## US           4.503096     28.506509
```

After carrying out Bagging, we can observe that the MSE is bettered to a value of 2.482. From the importance function we can also see that Price, ShelveLoc, CompPrice, Advertising and Age are the most important predictors for the response variable, sale.

```
#After removing variables

randf_seats = randomForest(Sales ~ ., data = Seats.train, mtry = 5, ntree = 310,
    importance = T)
randf_predict = predict(randf_seats, Seats.test)

MSE.removal<- mean((Seats.test$Sales - randf_predict)^2)
MSE.removal
```

```
## [1] 2.790671
```

```
importance(randf_seats)
```

```
##               %IncMSE IncNodePurity
## CompPrice   12.918772     141.89961
## Income       1.501915      92.36540
## Advertising 12.444714     135.33046
## Population   3.964158     117.08503
## Price       33.358484     437.95188
## ShelveLoc   28.944912     286.18506
## Age          7.657053     145.66025
## Education    3.415328      69.64734
## Urban       -1.100707      10.44787
## US           4.945748      38.87000
```

From this function we can see that Price, ShelveLoc, Advertising, CompPrice, and Age are still the most important predictorsFrom the reiterated run, we see that random forest worsens the MSE on test set to 2.707 as opposed to the previous 2.482.