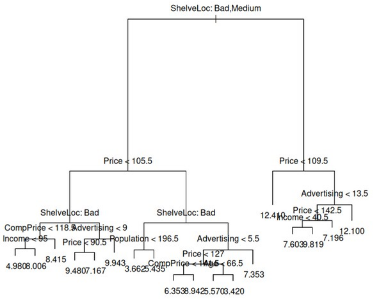
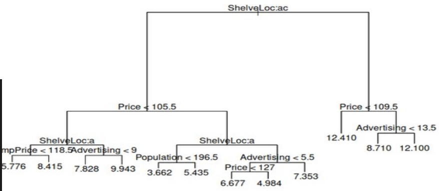
Question 8

a) I split the data into a test and training set at a 7-train to 1-test ratio. This gave me 357 training observations and 43 testing.

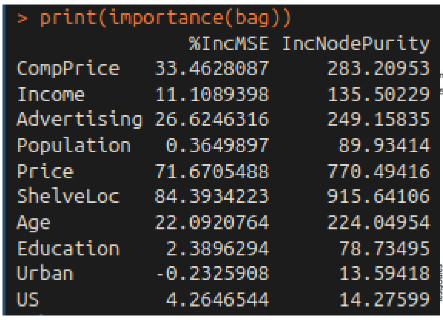
b) I created a tree and plotted it. It’s a little messy, but there’s a lot going on. I got a test MSE of 4.64



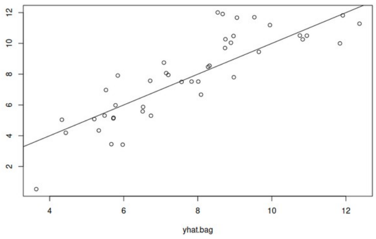
c) Pruning the tree using cross validation gave me a tree of size 12 terminal nodes instead of the initial 18 I had. I got a test MSE of 5.16, which actually performed worse. This makes sense in a regression setting, since trees are typically not the greatest at low observation regression



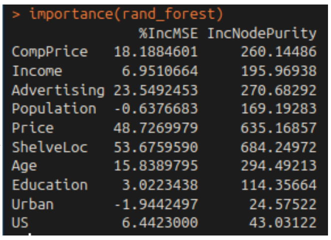
d) I bagged and plotted. The test MSE is 2.08. This is a significant decrease over the tree and pruned tree, which is a really good thing. After using importance(), the ShelveLoc was so extraordinarily important at a %IncMSE of 84, then Price at 71%, then CompPrice at 33, then Advertising 26%, and Age 22%.



```
> print(importance(bag))
                %IncMSE IncNodePurity
CompPrice    33.4628087     283.20953
Income       11.1089398     135.50229
Advertising  26.6246316     249.15835
Population    0.3649897      89.93414
Price        71.6705488     770.49416
ShelveLoc    84.3934223     915.64106
Age          22.0920764     224.04954
Education     2.3896294      78.73495
Urban        -0.2325908      13.59418
US            4.2646544      14.27599
```



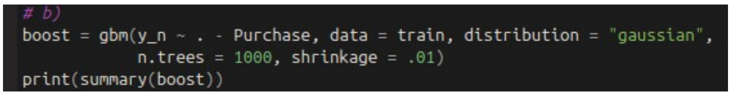
e) By using a random forest, I got a test MSE of 2.53, slightly worse than the bagged model. I chose an m of 3, which is roughly the square root of 10, the number of predictors used for the model. Here are the importances. According to the summary for the random forest model, mtry is 1 meaning it built a tree using only stumps, or an additive model. If m were larger, it might perform a little better, but then the results would be more correlated with each other since similar trees would get built.



```
> importance(rand_forest)
                %IncMSE IncNodePurity
CompPrice    18.1884601     260.14486
Income        6.9510664     195.96938
Advertising  23.5492453     270.68292
Population   -0.6376683     169.19283
Price        48.7269979     635.16857
ShelveLoc    53.6759590     684.24972
Age          15.8389795     294.49213
Education     3.0223438     114.35664
Urban        -1.9442497      24.57522
US            6.4423000      43.03122
```

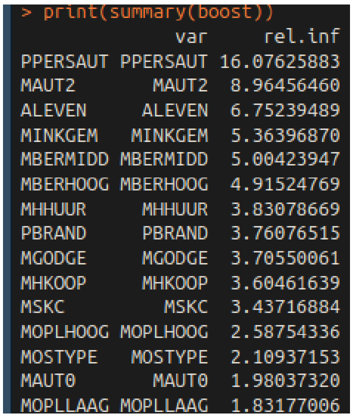
Question 11

a) I created a test and training set using rand



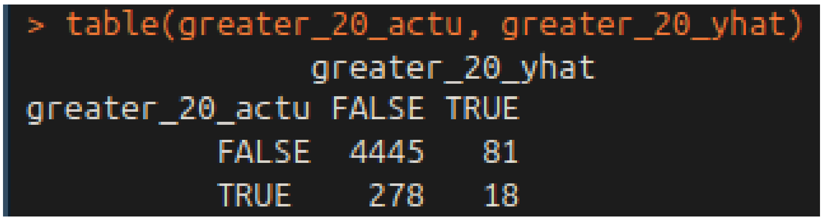
```
# b)
boost = gbm(y_n ~ . - Purchase, data = train, distribution = "gaussian",
            n.trees = 1000, shrinkage = .01)
print(summary(boost))
```

b) Here are the first few important predictors.



```
> print(summary(boost))
              var   rel.inf
PPERSAUT PPERSAUT 16.07625883
MAUT2       MAUT2  8.96456460
ALEVEN     ALEVEN  6.75239489
MINKGEM   MINKGEM  5.36396870
MBERMIDD MBERMIDD  5.00423947
MBERHOOG MBERHOOG  4.91524769
MHHUUR     MHHUUR  3.83078669
PBRAND     PBRAND  3.76076515
MGODGE     MGODGE  3.70550061
MHKOOP     MHKOOP  3.60461639
MSKC         MSKC  3.43716884
MOPLHOOG MOPLHOOG  2.58754336
MOSTYPE   MOSTYPE  2.10937153
MAUT0       MAUT0  1.98037320
MOPLLAAG MOPLLAAG  1.83177006
```

c) Based on this table, there are 99 observations that are predicted to be true, and only 18 of them are. This is a pretty bad rate, only 18% correct.



```
> table(greater_20_actu, greater_20_yhat)
               greater_20_yhat
greater_20_actu FALSE TRUE
          FALSE  4445   81
          TRUE    278   18
```