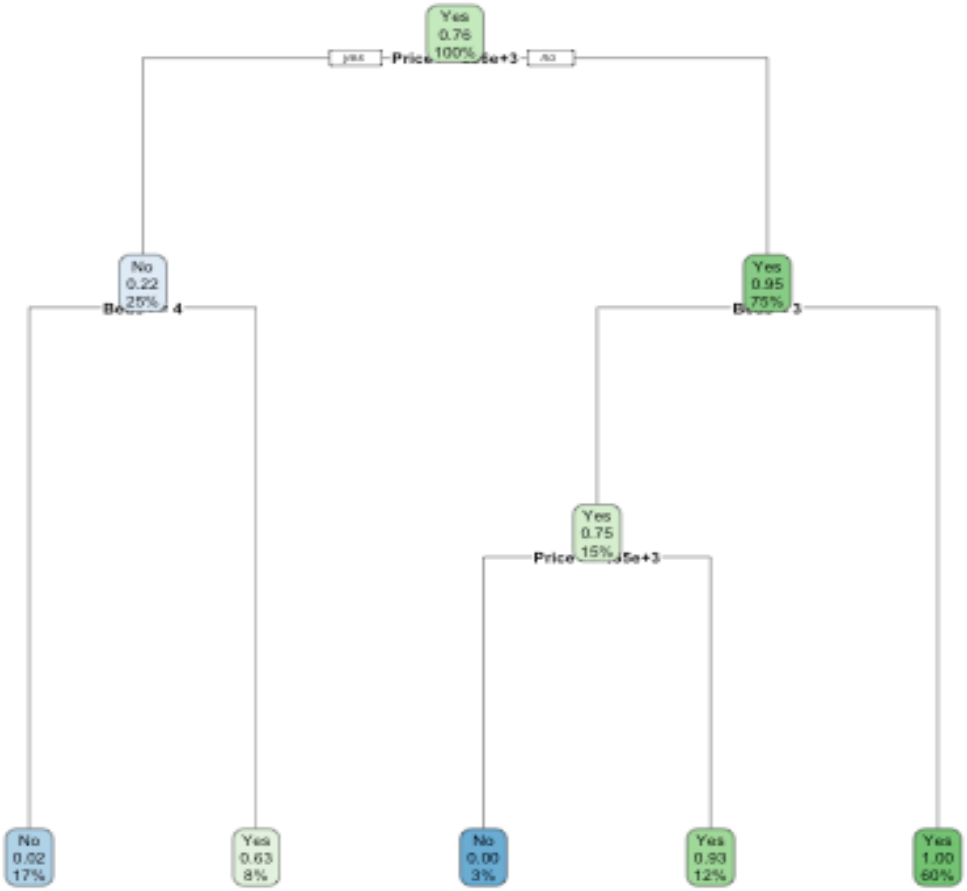
Swati Chauhan
MSTM ADV

TREE PRUNING

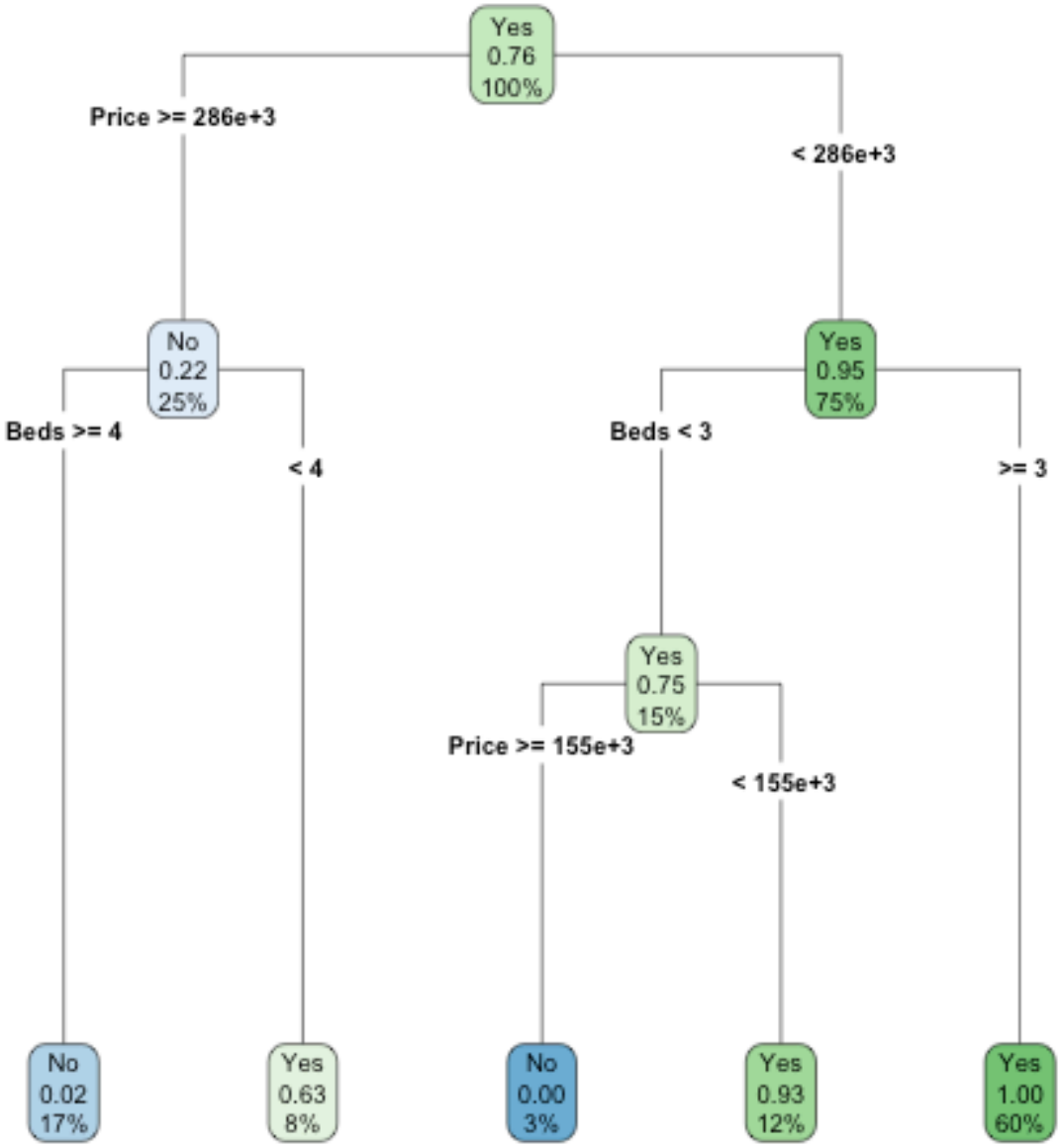
Use “invest” dataset to build a decision tree, target variable “Buy”. Use 80% of the data as training set. Please set.seed(50)

1. Report the decision tree graph (before and after pruning)



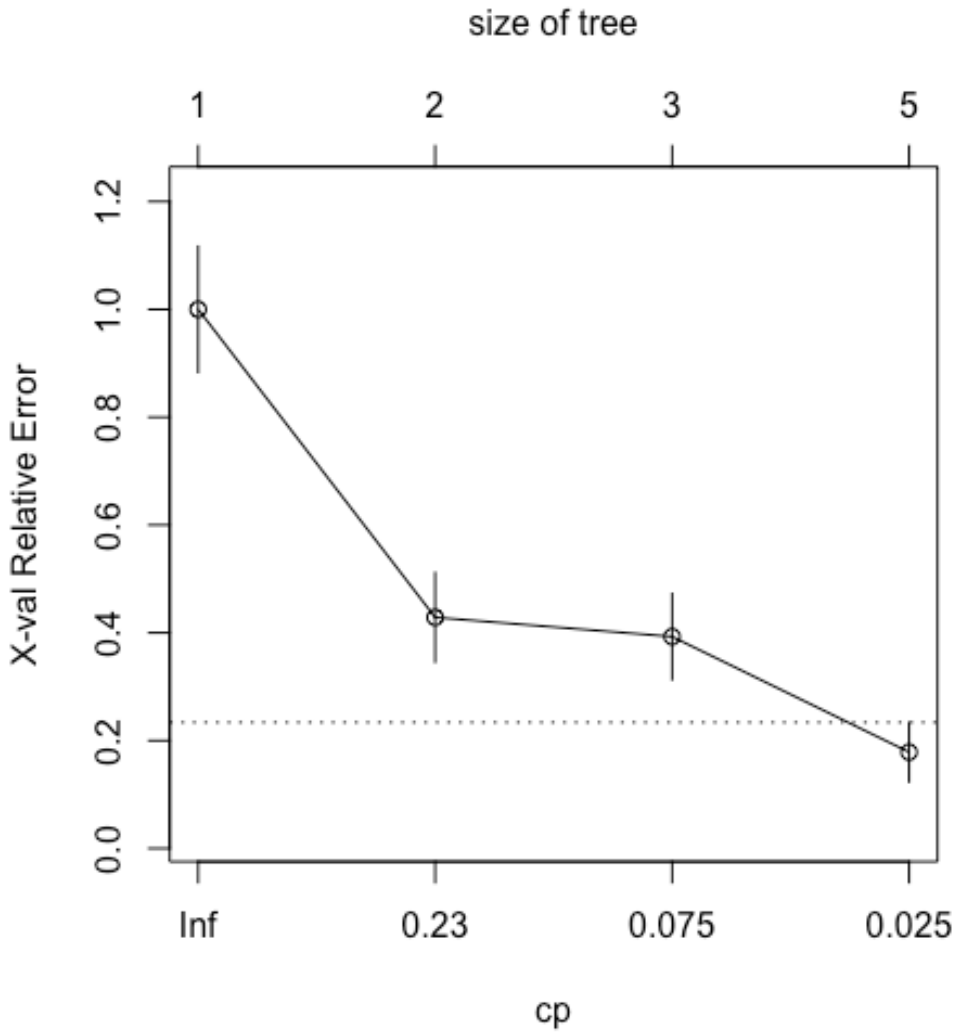
Swati Chauhan
MSTM ADV

TREE PRUNING



Swati Chauhan
MSTM ADV

TREE PRUNING



2) Write down the decision rules (after pruning) in plain English

Since our target variable is BUY 25% people will opt for price>286e+3 and 75% will opt for price<2863+e.Out of those 25% ,17% are not likely to buy beds>4 and 8% are likely to buy beds<4.For the rest 75% of the people,15% sun categorized into 3% who don't want to pay price>155e+3 and the rest 12% are ready to pay price<155e+3.The rest 60% are ready to buy with beds>=3 and are ready to pay any price<2863e+e

Swati Chauhan
MSTM ADV

TREE PRUNING

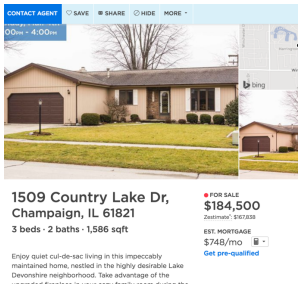
3. Report the accuracy, sensitivity and specificity of the tree (after pruning, using testing set)

```
predict_tree
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
28 29 30
 No  No  No Yes Yes  No Yes Yes Yes Yes Yes  No Yes Yes Yes Yes Yes  No Yes Yes Yes Yes
Yes  No  No Yes  No Yes Yes
 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55
56 57 58
Yes  No Yes  No Yes Yes Yes Yes  No Yes  No Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes
Yes Yes Yes Yes Yes
Levels: No Yes

predict_tree No Yes
       No  13   0
       Yes  0  45

predict_tree
     No Yes
 No  13   0
 Yes  0  45
```

4. Predict whether you should buy this property or not (use pruned tree):
When use predict() function, you can first put the information of the house into a dataframe:
a=data.frame(`Price`=xxx,`Baths`=x,`Beds`=x,`SQFT`=xxx,`Year`=1981)
https://www.zillow.com/homes/for_sale/house_type/3232892_zpid/40.138662,-88.22854,40.06303,-88.376169_rect/12_zm/




```
predict_new
 1
Yes
Levels: No Yes

> head(result,1)
  Price Baths Beds SQFT Year predict_new
1 184500     2    3 1586 1981        Yes
>
```

Swati Chauhan
MSTM ADV

TREE PRUNING

Final Script

```
View(invest)
summary(invest)
install.packages("caret")
library(caret)
set.seed(50)
trainIndex = createDataPartition(invest$Buy, p=0.8, list=FALSE,times=1 )
trainIndex
trainset = invest[trainIndex, ]
testset = invest[-trainIndex, ]
View(trainset)
View(testset)
library(rpart)
library(rpart.plot)
tree = rpart (Buy ~ ., trainset, method="class")
plot(tree)
text(tree)
tree
rpart.plot (tree,type=4,cex=.4)
rpart.plot(tree,type=2,cex=.4)
predict_tree = predict (tree, testset, type="class")
predict_tree
table (predict_tree)
table (predict_tree,testset$Buy)
table (testset$Buy,predict_tree)
printcp (tree)
plotcp (tree)
ptree = prune (tree, cp=tree$Buy[which.min(tree$cptable[,"xerror"]), "CP"])
rpart.plot(ptree,type=4,cex=.6)
predict_ptree=predict(ptree,testset,type="class")
table (predict_ptree,testset$Buy)
predict(ptree,testset[1,],type="class")
a=data.frame(`Price`=184500,`Baths`=2,`Beds`=3,`SQFT`=1586,`Year`=1981)
predict_new = predict(tree, a , type="class")
predict_new
result= cbind(a,predict_new)
View(result)
```

Swati Chauhan
MSTM ADV

TREE PRUNING

Final Outputs

```
n= 238

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 238 56 Yes (0.23529412 0.76470588)
   2) Price>=286450 60 13 No (0.78333333 0.21666667)
     4) Beds>=3.5 41  1 No (0.97560976 0.02439024) *
     5) Beds< 3.5 19  7 Yes (0.36842105 0.63157895) *
   3) Price< 286450 178  9 Yes (0.05056180 0.94943820)
     6) Beds< 2.5 36  9 Yes (0.25000000 0.75000000)
      12) Price>=154950 7  0 No (1.00000000 0.00000000) *
      13) Price< 154950 29  2 Yes (0.06896552 0.93103448) *
     7) Beds>=2.5 142  0 Yes (0.00000000 1.00000000) *

Classification tree:
rpart(formula = Buy ~ ., data = trainset, method = "class")

Variables actually used in tree construction:
[1] Beds  Price

Root node error: 56/238 = 0.23529

n= 238

        CP nsplit rel error  xerror     xstd
1 0.607143      0   1.00000 1.00000 0.116857
2 0.089286      1   0.39286 0.42857 0.082954
3 0.062500      2   0.30357 0.39286 0.079792
4 0.010000      4   0.17857 0.17857 0.055270
```