CART

Sumera Saleem

8/28/2020

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------

## v ggplot2 3.3.1     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------------------------------ tid
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dummies)
```

```
## Warning: package 'dummies' was built under R version 4.0.2

## dummies-1.5.6 provided by Decision Patterns
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.2

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.0.2
```

```
library("tree")
```

```
## Warning: package 'tree' was built under R version 4.0.2
```

```
## Registered S3 method overwritten by 'tree':
##   method     from
##   print.tree cli
```

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.0.2
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

Data Preprocessing. Convert variable Duration into a categorical variable. Split the data into training (60%) and validation (40%) datasets.

```
getwd
```

```
## function ()
## .Internal(getwd())
## <bytecode: 0x7fc501849458>
## <environment: namespace:base>
```

```
ebay<-read.csv("/Users/sumerasaleem/Desktop/Data mining/dmba/eBayAuctions.csv")
#Upload data
str(ebay)
```

```
## 'data.frame':    1972 obs. of  8 variables:
##  $ Category    : chr  "Music/Movie/Game" "Music/Movie/Game" "Music/Movie/Game" "Music/Movie/Game" ..
##  $ currency    : chr  "US" "US" "US" "US" ...
##  $ sellerRating: int  3249 3249 3249 3249 3249 3249 3249 3249 3249 3249 ...
##  $ Duration    : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ endDay      : chr  "Mon" "Mon" "Mon" "Mon" ...
##  $ ClosePrice  : num  0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 ...
##  $ OpenPrice   : num  0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 ...
##  $ Competitive.: int  0 0 0 0 0 0 0 0 0 0 ...
```

```
# check structure of data
ebay<- ebay%>% mutate_if(is.character,as.factor)
#Make sure all the categorical variables are converted into factors.
ebay$Duration=as.factor(ebay$Duration)
#convert Duration variable type into factor type
ebay$Competitive.=as.factor(ebay$Competitive.)
#convert Competitive variable type into factor type
set.seed(100)
train.index<-sample(nrow(ebay),nrow(ebay)*0.6)
# Partion data into 60/40. takes 60% row into train.index
```

```
valid.index<-setdiff(rownames(ebay),train.index)
#40 % into valid.index
train.data<-ebay[train.index,]
head(train.data)
```

```
##                  Category currency sellerRating Duration endDay ClosePrice
## 1738        SportingGoods       US           62        5    Mon      66.00
## 503      Music/Movie/Game      GBP          105        7    Thu       5.71
## 1382 Clothing/Accessories      EUR          164        7    Tue      30.62
## 1648         Toys/Hobbies       US          534        7    Sun      19.95
## 985     Antique/Art/Craft       US         4390        7    Mon       6.49
## 1742         Toys/Hobbies       US           88        5    Mon      71.00
##      OpenPrice Competitive.
## 1738      9.99            1
## 503       4.44            1
## 1382      6.03            0
## 1648     19.95            0
## 985       6.49            0
## 1742      9.99            1
```

```
# transfer values into of rows from train.index and all col to train.data
valid.data<-ebay[valid.index,]
head(valid.data)
```

```
##            Category currency sellerRating Duration endDay ClosePrice OpenPrice
## 5  Music/Movie/Game       US         3249        5    Mon       0.01      0.01
## 7  Music/Movie/Game       US         3249        5    Mon       0.01      0.01
## 8  Music/Movie/Game       US         3249        5    Mon       0.01      0.01
## 9  Music/Movie/Game       US         3249        5    Mon       0.01      0.01
## 10 Music/Movie/Game       US         3249        5    Mon       0.01      0.01
## 14 Music/Movie/Game       US         3249        5    Mon       0.01      0.01
##    Competitive.
## 5             0
## 7             0
## 8             0
## 9             0
## 10            0
## 14            0
```

```
# transfer values into of rows from valid.index and all col to valid.data
```

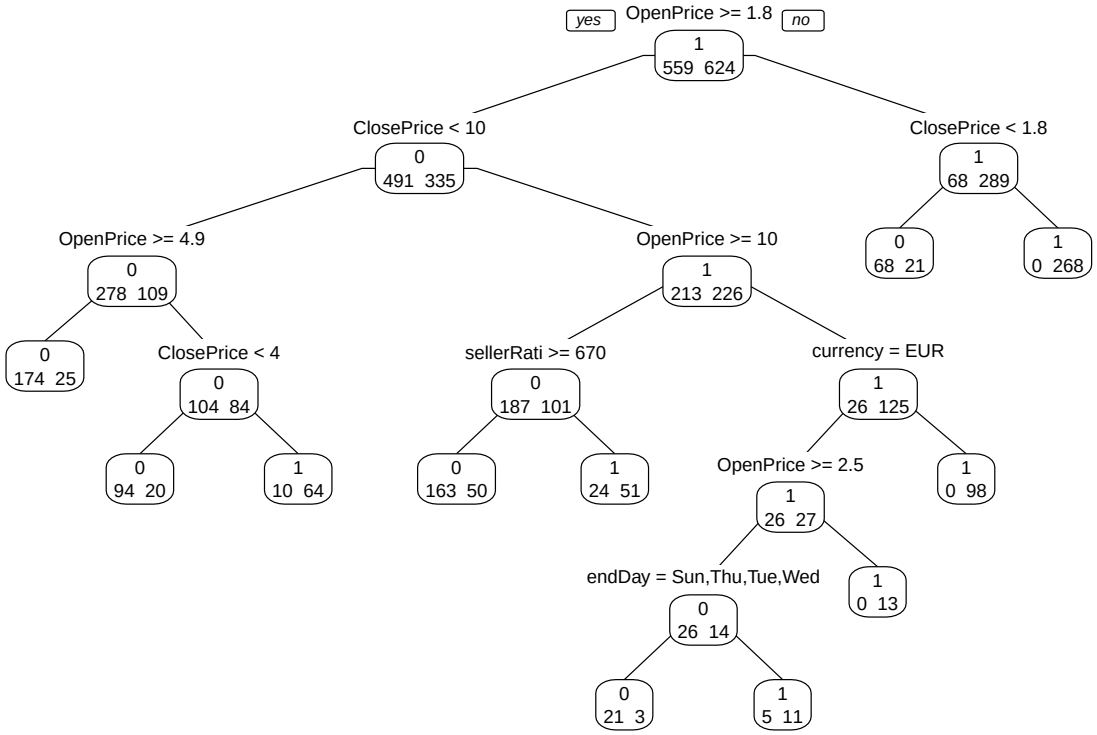
Fit a classification tree using all predictors, using the best-pruned tree. To avoid overfitting, set the minimum number of records in a terminal node to 50 (in R: minbucket = 50). Also, set the maximum number of levels to be displayed at seven (in R: maxdepth = 7).Write down the results in terms of rules. (Note: If you had to slightly reduce the number of predictors due to software limitations, or for clarity of presentation, which would be a good variable to choose?)

```
t(t(names(valid.data)))
```

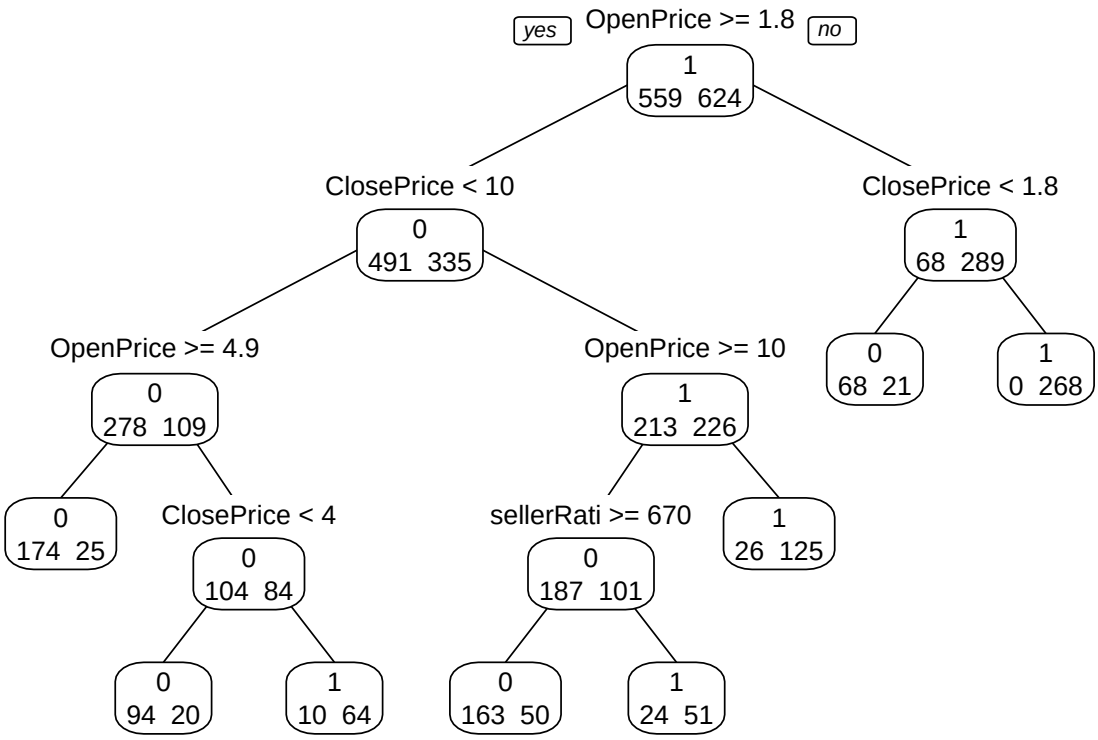
```
##      [,1]
## [1,] "Category"
```

```
## [2,] "currency"
## [3,] "sellerRating"
## [4,] "Duration"
## [5,] "endDay"
## [6,] "ClosePrice"
## [7,] "OpenPrice"
## [8,] "Competitive."
```

```
default.ct<-rpart(Competitive. ~ .,data=train.data,method="class")
# first of all grow basic Classification tree.
prp(default.ct,type=1,extra=1,split.font = 1,varlen=-10)
```




```
# plot tree
#printcp(default.ct)
#default.ct.pred<-predict(default.ct,train.data,type="class")
#confusionMatrix(default.ct.pred,train.data$Competitive.)
default.ct1<-rpart(Competitive.~.,data=train.data,
                   control=rpart.control(maxdepth=7,minsplit = 5,minbucket= 50,xval = 5),
                   method = "class",cp = 0.00001)
# grow
prp(default.ct1,type=1,extra=1,split.font = 1,varlen=-10)
```



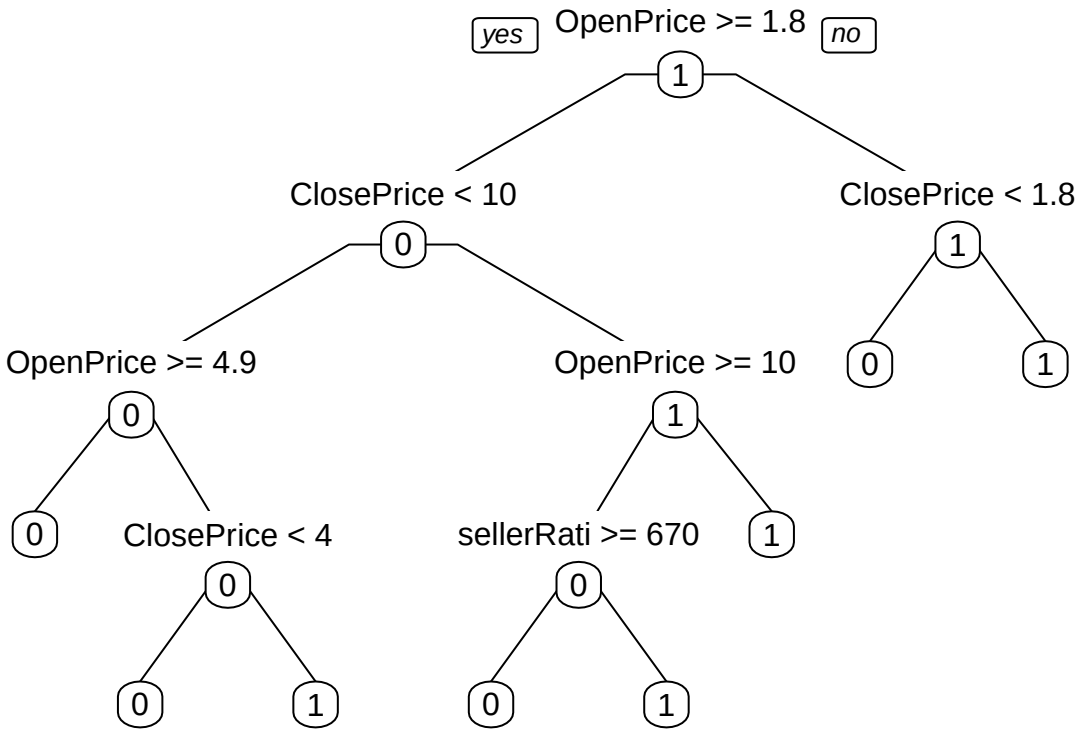

```
printcp(default.ct1)
```

```
##
## Classification tree:
## rpart(formula = Competitive. ~ ., data = train.data, method = "class",
##     control = rpart.control(maxdepth = 7, minsplit = 5, minbucket = 50,
##         xval = 5), cp = 1e-05)
##
## Variables actually used in tree construction:
## [1] ClosePrice   OpenPrice    sellerRating
##
## Root node error: 559/1183 = 0.47253
##
## n= 1183
##
##         CP nsplit rel error  xerror     xstd
## 1 0.279070      0   1.00000 1.00000 0.030718
## 2 0.088551      1   0.72093 0.73524 0.029297
## 3 0.084079      3   0.54383 0.60465 0.027796
## 4 0.048301      4   0.45975 0.50447 0.026217
## 5 0.010000      7   0.31485 0.40787 0.024270
```

```
best.pruned.tree<-prune(default.ct1,cp=default.ct1$cptable[which.min(default.ct1$cptable[,"xerror"]),"CP
length(best.pruned.tree$frame$var[best.pruned.tree=="<leaf>"])
```

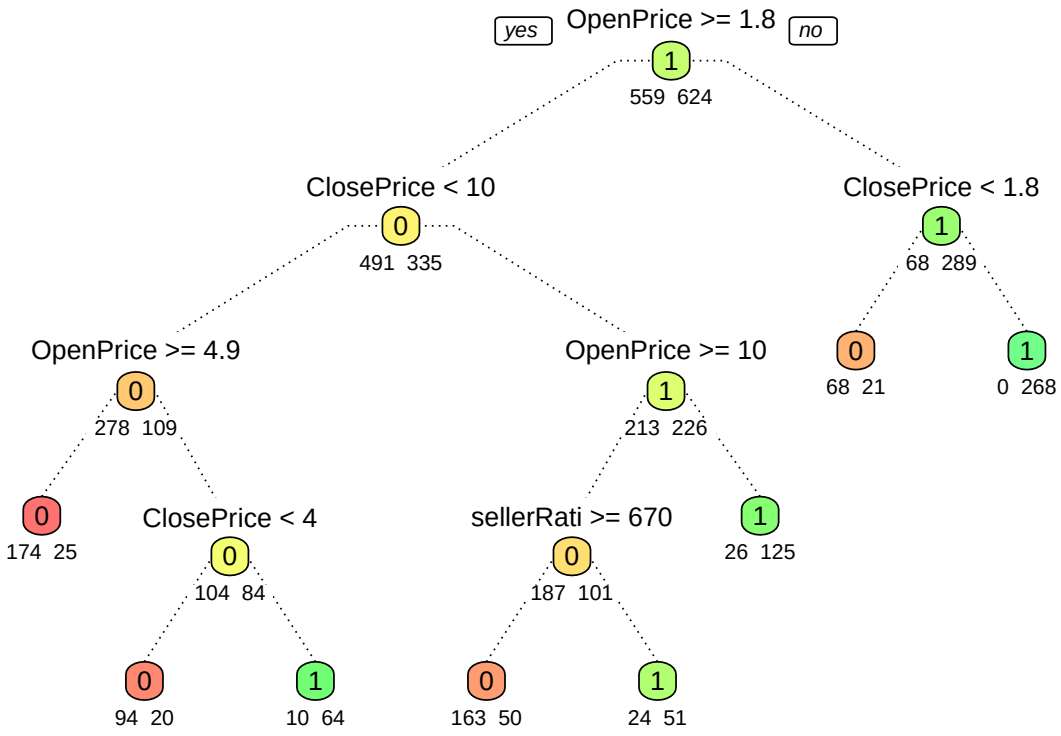
```
## [1] 0
```

```
prp(best.pruned.tree,type = 1,split.font = 1,varlen = -10)
```



Fit another classification tree (using the best-pruned tree, with a minimum number of records per terminal node = 50 and maximum allowed number of displayed levels = 7), this time only with predictors that can be used for predicting the outcome of a new auction. Describe the resulting tree in terms of rules. Make sure to report the smallest set of rules required for classification.

```
new.ct<-rpart(Competitive.~ClosePrice+OpenPrice+sellerRating,data=train.data,
                  control=rpart.control(maxdepth=7,minsplit = 5),
                  minbucket= 50,method = "class")
prp(new.ct,type=1,extra=1,under = TRUE,split.font = 1,
    varlen=-10,branch.lty = 3, box.palette = "RdYlGn")
```




```
printcp(new.ct)
```

```
##
## Classification tree:
## rpart(formula = Competitive. ~ ClosePrice + OpenPrice + sellerRating,
##     data = train.data, method = "class", control = rpart.control(maxdepth = 7,
##         minsplit = 5), minbucket = 50)
##
## Variables actually used in tree construction:
## [1] ClosePrice   OpenPrice    sellerRating
##
## Root node error: 559/1183 = 0.47253
##
## n= 1183
##
##         CP nsplit rel error  xerror     xstd
## 1 0.279070      0   1.00000 1.00000 0.030718
## 2 0.088551      1   0.72093 0.72630 0.029213
## 3 0.084079      3   0.54383 0.61002 0.027869
## 4 0.048301      4   0.45975 0.47764 0.025722
## 5 0.010000      7   0.31485 0.33453 0.022446
```

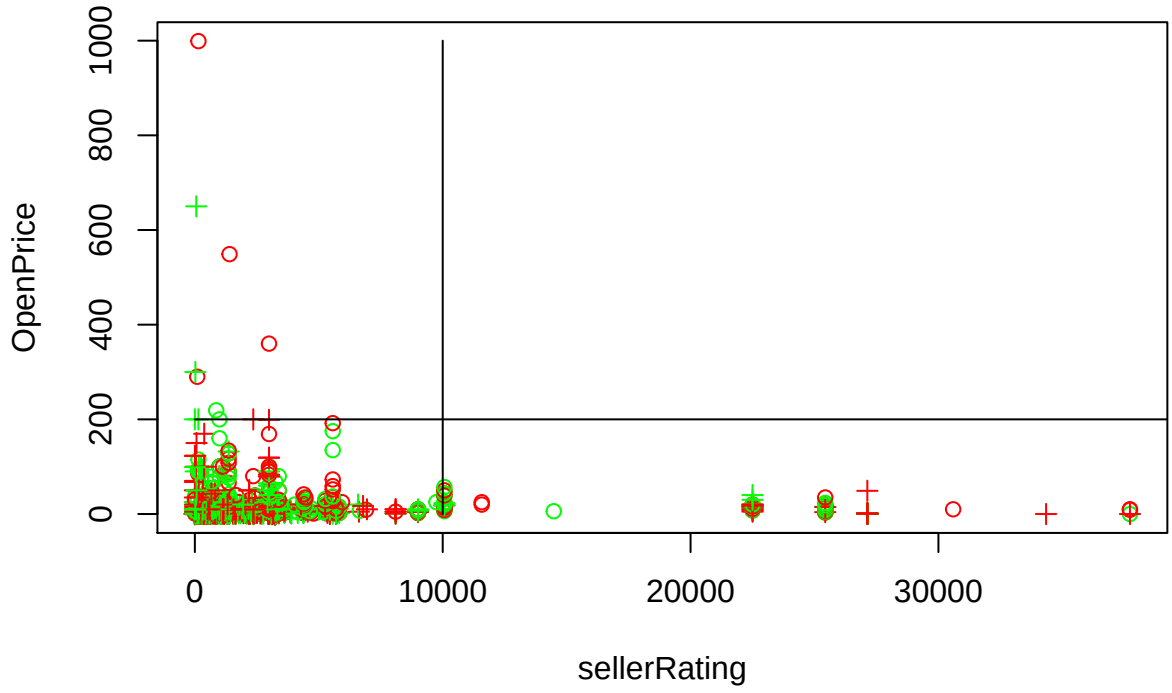
Plot the resulting tree on a scatter plot: Use the two axes for the two best (quantitative) predictors. Each auction will appear as a point, with coordinates corresponding to its values on those two predictors. Use different colors or symbols to separate competitive and noncompetitive auctions. Draw lines (you can sketch

these by hand or use R) at the values that create splits. Does this splitting seem reasonable with respect to the meaning of the two predictors? Does it seem to do a good job of separating the two classes?

```
str(ebay)
```

```
## 'data.frame':    1972 obs. of  8 variables:
##  $ Category    : Factor w/ 18 levels "Antique/Art/Craft",..: 14 14 14 14 14 14 14 14 14 14 ...
##  $ currency    : Factor w/ 3 levels "EUR","GBP","US": 3 3 3 3 3 3 3 3 3 3 ...
##  $ sellerRating: int  3249 3249 3249 3249 3249 3249 3249 3249 3249 3249 ...
##  $ Duration    : Factor w/ 5 levels "1","3","5","7",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ endDay      : Factor w/ 7 levels "Fri","Mon","Sat",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ ClosePrice  : num  0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 ...
##  $ OpenPrice   : num  0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 ...
##  $ Competitive.: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

```
plot(OpenPrice~sellerRating ,data=ebay,pch=ifelse(ebay$Competitive.==0,1,3),
col=c("green","red"))+lines(c(0,40000),c(200,200))+lines(c(10000,10000),c(0,1000))
```




```
## integer(0)
```

Predicting Prices of Used Cars (Regression Trees). The file ToyotaCorolla.csv contains the data on used cars (Toyota Corolla) on sale during late summer of 2004 in the Netherlands. It has 1436 records containing details on 38 attributes, including Price, Age, Kilometers, HP, and other specifications. The goal is to predict the price of a used Toyota Corolla based on its specifications. (The example in Section 9.7 is a subset of this dataset). Data Preprocessing. Split the data into training (60%), and validation (40%) datasets.

```
ToyotaCorolla<-read.csv("/Users/sumerasaleem/Desktop/Data mining/dmba/ToyotaCorolla.csv")
set.seed(22)
rows.index<-sample(nrow(ToyotaCorolla),nrow(ToyotaCorolla)*0.6)
train.data<-ToyotaCorolla[rows.index,]
head(train.data)
```

```
##        Id                                                Model Price Age_08_04
## 393   395 TOYOTA Corolla 1.6 16V VVT I WAGON TERRA Stationwagen  9950        49
## 1354 1360            TOYOTA Corolla 1.6 LB LINEA LUNA 4/5-Doors  8250        80
## 300   301      TOYOTA Corolla 1.6 16V VVT I HATCHB G6 2/3-Doors 13750        39
## 1404 1410               TOYOTA Corolla 1.3 16V HATCHB 2/3-Doors  7000        73
## 1339 1345  TOYOTA Corolla Hatchback 1.6 Terra Comfort 2/3-Doors  7499        80
## 721   724    TOYOTA Corolla 1.6 16V LIFTB LINEA TERRA 4/5-Doors  8450        63
##      Mfg_Month Mfg_Year     KM Fuel_Type  HP Met_Color Color Automatic   CC
## 393          8     2000 131364    Petrol 110         1 Black         0 1600
## 1354         1     1998  60476    Petrol 110         1  Grey         0 1600
## 300          6     2001  40000    Petrol 110         1  Grey         0 1600
## 1404         8     1998  47360    Petrol  86         0  Grey         0 1300
## 1339         1     1998  63500    Petrol 110         1  Blue         0 1600
## 721          6     1999  88685    Petrol 110         0  Grey         0 1600
##      Doors Cylinders Gears Quarterly_Tax Weight Mfr_Guarantee BOVAG_Guarantee
## 393      5         4     5            85   1075             0               1
## 1354     5         4     5            19   1114             0               1
## 300      3         4     5            85   1055             0               1
## 1404     3         4     5            69   1010             0               1
## 1339     3         4     5            69   1050             0               0
## 721      5         4     5            85   1070             1               1
##      Guarantee_Period ABS Airbag_1 Airbag_2 Airco Automatic_airco Boardcomputer
## 393                 3   1        1        1     0              0             1
## 1354                3   1        1        0     1              1             0
## 300                 3   1        1        1     1              0             1
## 1404                3   0        1        0     0              0             0
## 1339                3   0        1        0     0              0             0
## 721                 3   1        1        1     0              0             0
##      CD_Player Central_Lock Powered_Windows Power_Steering Radio Mistlamps
## 393          0            1               1              1     0         0
## 1354         0            1               1              1     0         1
## 300          0            1               1              1     0         1
## 1404         0            0               0              1     0         0
## 1339         0            1               1              1     0         0
## 721          0            0               0              1     0         0
##      Sport_Model Backseat_Divider Metallic_Rim Radio_cassette Parking_Assistant
## 393            0               1            0              0                 0
## 1354           0               0            0              0                 0
## 300            0               1            1              0                 0
## 1404           0               1            0              0                 0
## 1339           0               0            0              0                 0
## 721            1               1            0              0                 0
##      Tow_Bar
## 393        0
## 1354       0
## 300        1
## 1404       0
```

```
## 1339       1
## 721        0
```

```
valid.data<-ToyotaCorolla[-rows.index,]
head(valid.data)
```

```
##    Id                                          Model Price Age_08_04
## 7   7      TOYOTA Corolla 2.0 D4D 90 3DR TERRA 2/3-Doors 16900        27
## 8   8      TOYOTA Corolla 2.0 D4D 90 3DR TERRA 2/3-Doors 18600        30
## 10 10        TOYOTA Corolla 1.9 D HATCHB TERRA 2/3-Doors 12950        23
## 13 13 TOYOTA Corolla 1.8 16V VVTLI 3DR T SPORT 2/3-Doors 19600        25
## 15 15 TOYOTA Corolla 1.8 16V VVTLI 3DR T SPORT 2/3-Doors 22500        32
## 16 16 TOYOTA Corolla 1.8 16V VVTLI 3DR T SPORT 2/3-Doors 22000        28
##    Mfg_Month Mfg_Year    KM Fuel_Type  HP Met_Color Color Automatic   CC Doors
## 7          6     2002 94612    Diesel  90         1  Grey         0 2000     3
## 8          3     2002 75889    Diesel  90         1  Grey         0 2000     3
## 10        10     2002 71138    Diesel  69         0  Blue         0 1900     3
## 13         8     2002 32189    Petrol 192         0   Red         0 1800     3
## 15         1     2002 34131    Petrol 192         1  Grey         0 1800     3
## 16         5     2002 18739    Petrol 192         0  Grey         0 1800     3
##    Cylinders Gears Quarterly_Tax Weight Mfr_Guarantee BOVAG_Guarantee
## 7          4     5           210   1245             0               1
## 8          4     5           210   1245             1               1
## 10         4     5           185   1105             0               1
## 13         4     6           100   1185             1               1
## 15         4     6           100   1185             1               1
## 16         4     6           100   1185             0               1
##    Guarantee_Period ABS Airbag_1 Airbag_2 Airco Automatic_airco Boardcomputer
## 7                 3   1        1        1     1               0             1
## 8                 3   1        1        1     1               0             1
## 10                3   1        1        1     1               0             1
## 13                3   1        1        1     1               1             1
## 15                3   1        1        1     1               1             1
## 16                3   1        1        1     1               1             1
##    CD_Player Central_Lock Powered_Windows Power_Steering Radio Mistlamps
## 7          0            1               1              1     0         0
## 8          1            1               1              1     0         0
## 10         0            0               0              1     0         0
## 13         0            1               1              1     0         1
## 15         1            1               1              1     0         1
## 16         0            1               1              1     0         1
##    Sport_Model Backseat_Divider Metallic_Rim Radio_cassette Parking_Assistant
## 7            1               1           0              0                 0
## 8            0               1           0              0                 0
## 10           0               1           0              0                 0
## 13           1               1           1              0                 0
## 15           1               1           1              0                 0
## 16           1               1           1              0                 0
##    Tow_Bar
## 7        0
## 8        0
## 10       0
## 13       0
## 15       0
```

```
## 16       0
```

a. Run a regression tree (RT) with outcome variable Price and predictors Age_08_04, KM, Fuel_Type, HP, Automatic, Doors, Quarterly_Tax, Mfg_Guarantee, Guarantee_Period, Airco, Automatic_Airco, CD_Player, Powered_Windows, Sport_Model, and Tow_Bar. Keep the minimum number of records in a terminal node to 1, maximum number of tree levels to 100, and cp = 0:001, to make the run least restrictive.

Note: Maxdepth of nodes are 30 in R. so we will use maxdepth as30 (i) Which appear to be the three or four most important car specifications for predicting the car's price? Answer: Age_08_04: 9988554538 KM:3204175660 Automatic_airco: 3062195648 Quarterly_Tax: 1577421742 Age ,KM Automatic_airco and Quarterly_tax are four top important variables for predicting price.

ii)Compare the prediction errors of the training and validation sets by examining their RMS error and by plotting the two boxplots. What is happening with the training set predictions? How does the predictive performance of the validation set compare to the training set? Why does this occur?

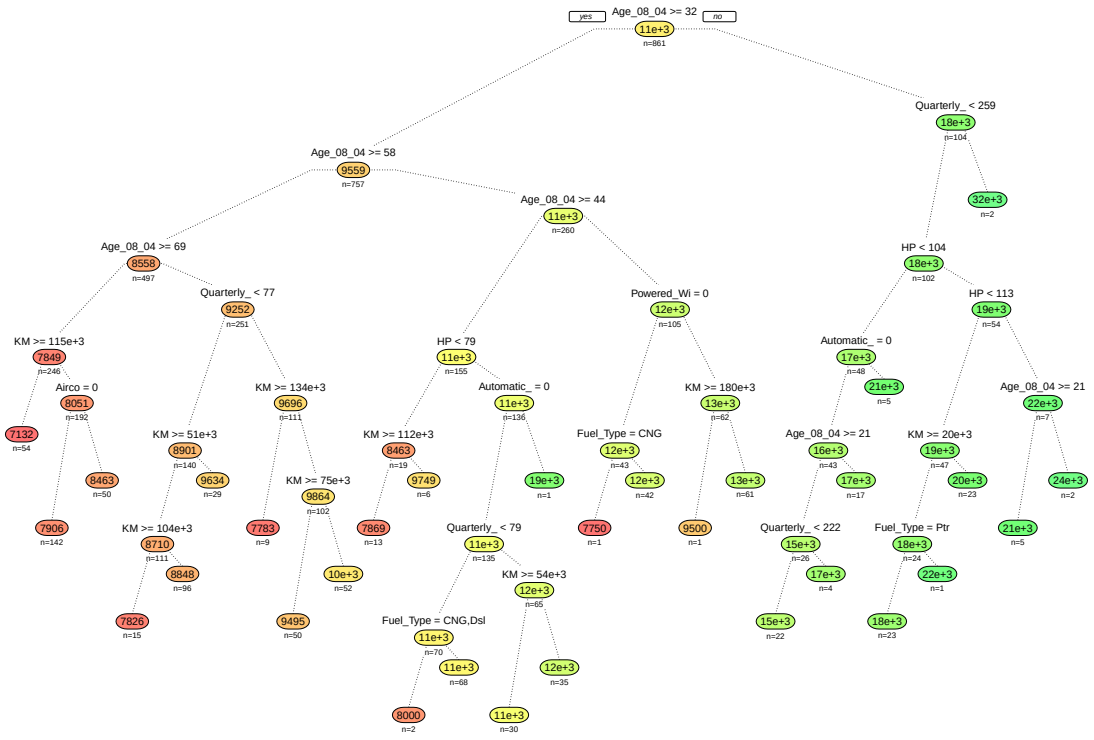
Answer: RMS error is low in train data as it has more no of rows and it was overfitted in camparison with valid data. from boxplot we can see with training data we can fewer outlier as it has more data and in validation set we see more outliers which proves it didnt overfit the data.

iii)How can we achieve predictions for the training set that are not equal to the actual prices? Answer: We can achive unequal predictions based on how how we have take taken training data set, if its close to actual data then predictions will be same.

iv)Prune the full tree using the cross-validation error. Compared to the full tree, what is the predictive performance for the validation set?

Answer: As expected, compared to the full tree, our pruned tree performs worse on the training set (RMSE=1343 compared to 993 for the full tree). The validation set also performed worse better(RMSE=1441 compared to 1319), but the pruned validation set performed better than the pruned training set. This indicates that we have underfit our model.

```
RT<-rpart(Price ~ Age_08_04+KM+Fuel_Type+HP+Automatic+Doors+
            Quarterly_Tax+ Mfr_Guarantee+Guarantee_Period+Airco+
            Automatic_airco+CD_Player+Powered_Windows+Sport_Model+Tow_Bar,
            data=train.data,
            method = "anova",
            minbucket=1,
            maxdepth=30,
            cp=0.001)
# Grow tree on given variables
vis_RT<-prp(RT,type=1,extra=1,under = TRUE,split.font = 1,varlen=-10,
            branch.lty = 3, box.palette = "RdYlGn")
```



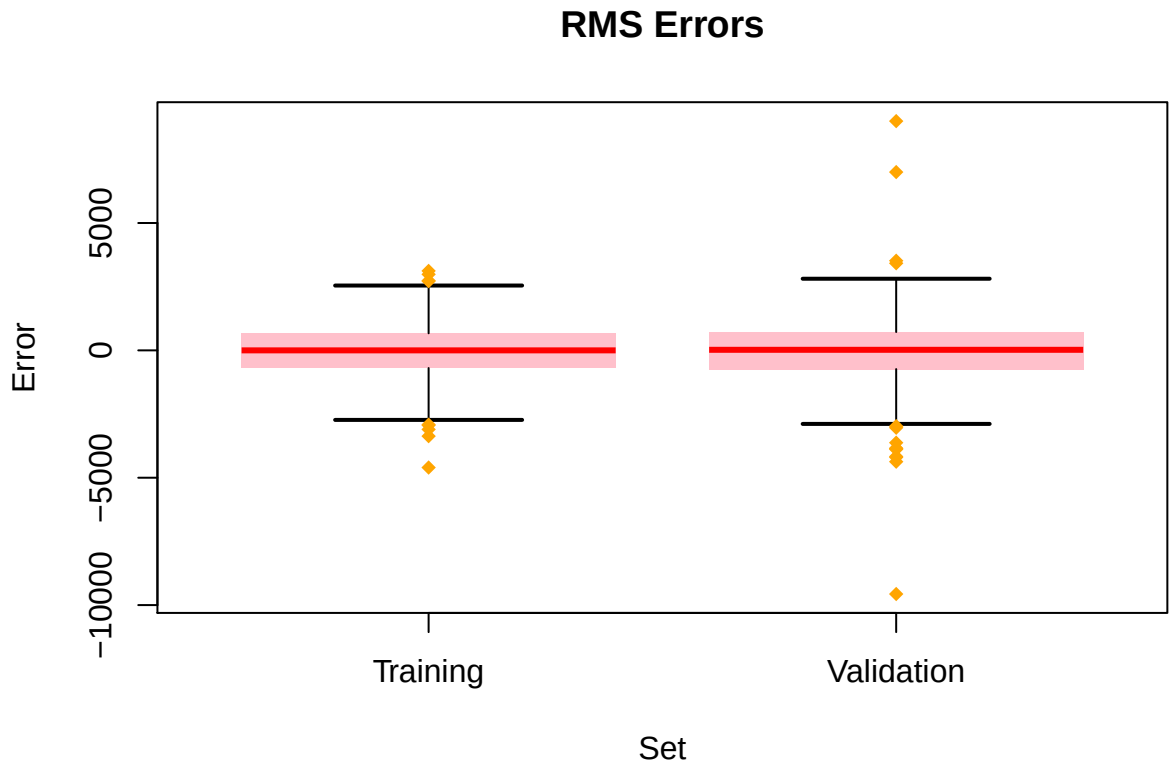

```
# its hard to visuals with tree grown with maxdepth
# names of varaibles of tree in descending importance level.
t(t(RT$variable.importance))
```

```
##                       [,1]
## Age_08_04        8883389342
## Automatic_airco  2919505875
## KM               2384942268
## Quarterly_Tax    1607048729
## HP               1050164969
## CD_Player         343306312
## Fuel_Type         235685722
## Guarantee_Period  208534738
## Airco             122851533
## Powered_Windows    64981981
## Doors              59287239
## Mfr_Guarantee      43522929
## Automatic           5272476
```

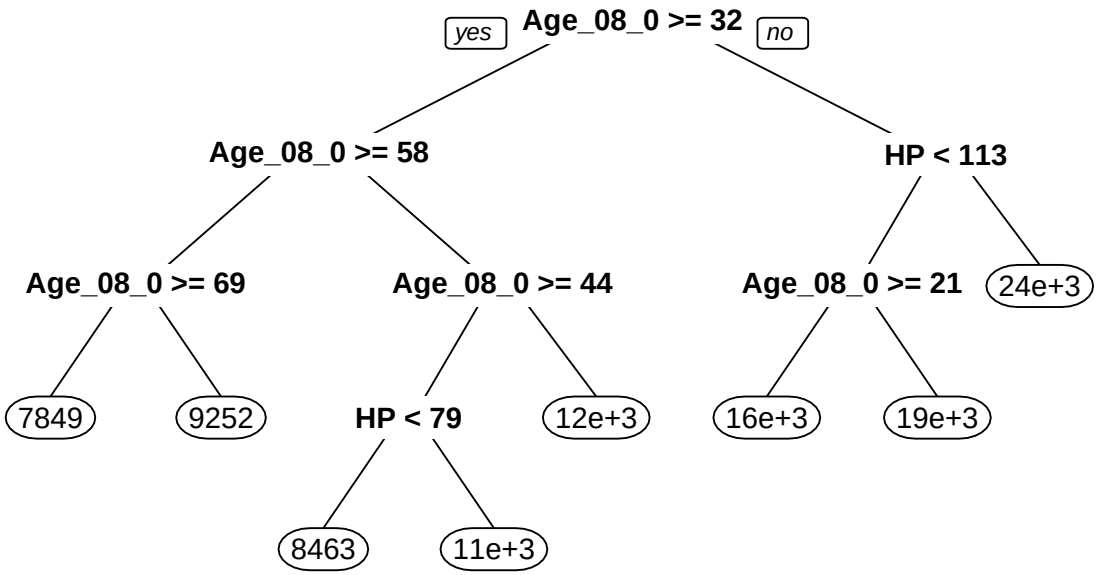
```
# names of varaibles of tree in descending importance level.
pred.price.train<-predict(RT,train.data)
pred.price.valid<-predict(RT,valid.data)
accuracy.train<-accuracy(pred.price.train,train.data$Price)
accuracy.valid<-accuracy(pred.price.valid,valid.data$Price)
# accuracy for both predicted values on train and valid data.
```

```r
train.err <-pred.price.train -train.data$Price
valid.err <-pred.price.valid -valid.data$Price

#to calculate the errors you need to subtract the prediction from the actual
#create a data fram from the training and validation errors in order to plot
err <-data.frame(Error =c(train.err, valid.err),
Set =c(rep("Training", length(train.err)),
rep("Validation", length(valid.err))))
#create your error box plots
boxplot(Error~Set, data=err, main="RMS Errors",
xlab = "Set", ylab = "Error",
col="pink",medcol="red",boxlty=0,border="black",
whisklty=1,staplelwd=2,outpch=18,outcex=1,outcol="orange")
```




```r
# for full tree there will be no control
full.tree<-rpart(Price ~ Age_08_04+KM+Fuel_Type+HP+Automatic+Doors+
            Quarterly_Tax+ Mfr_Guarantee+Guarantee_Period+Airco+
            Automatic_airco+CD_Player+Powered_Windows+Sport_Model+Tow_Bar,
            data=train.data)
prp(full.tree)
```




```
pred.newtreeprice.train<-predict(full.tree,train.data)
pred.newtreeprice.valid<-predict(full.tree,valid.data)
accuracy.newtree.train<-accuracy(pred.newtreeprice.train,train.data$Price)
accuracy.newtree.valid<-accuracy(pred.newtreeprice.valid,valid.data$Price)
accuracy.newtree.train
```

```
##                      ME     RMSE      MAE      MPE     MAPE
## Test set 3.422132e-13 1342.992 1006.305 -1.64915 10.05497
```

```
accuracy.newtree.valid
```

```
##                ME     RMSE      MAE        MPE     MAPE
## Test set 65.81223 1440.945 1043.952 -0.8139846 9.827839
```

b)Let us see the effect of turning the price variable into a categorical variable. First, create a new variable that categorizes price into 20 bins. Now repartition the data keeping Binned_Price instead of Price. Run a classification tree with the same set of input variables as in the RT, and with Binned_Price as the output variable. Keep the minimum number of records in a terminal node to 1. Answer: To start with this question first we need to create bin for Price variable.

```
#create bins based on the car prices as categories
#determine the number of bins based on the min and max prices
bins <- seq(min(ToyotaCorolla$Price),
max(ToyotaCorolla$Price),
(max(ToyotaCorolla$Price) - min(ToyotaCorolla$Price))/20)
bins
```

```
##  [1]  4350.0  5757.5  7165.0  8572.5  9980.0 11387.5 12795.0 14202.5 15610.0
## [10] 17017.5 18425.0 19832.5 21240.0 22647.5 24055.0 25462.5 26870.0 28277.5
## [19] 29685.0 31092.5 32500.0
```

```
Binned_Price <- .bincode(ToyotaCorolla$Price,
                         bins,
                         include.lowest = TRUE)
#convert the Binned_Price to factors for classification
Binned_Price <- as.factor(Binned_Price)
head(Binned_Price)
```

```
## [1] 7 7 7 8 7 7
## Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 19 20
```

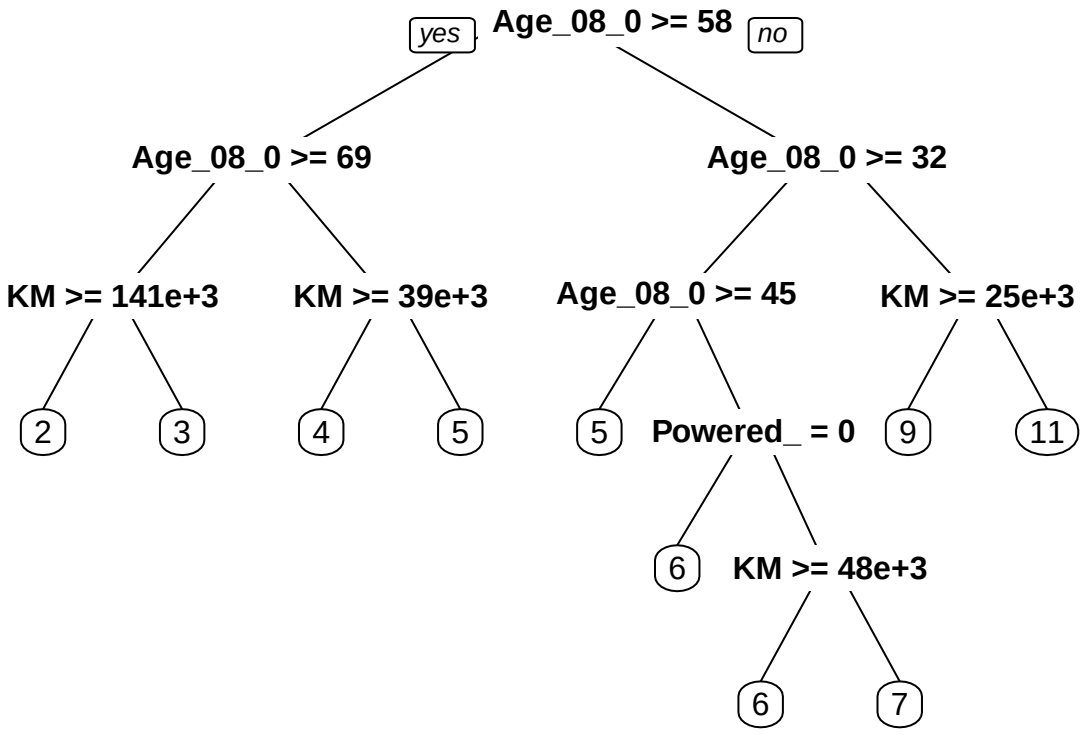
```
#add the binned price to the data frame based on the indexes
#this will allow us to identify the training and validation data frames


train.data$Binned_Price <- Binned_Price[rows.index]
valid.data$Binned_Price <- Binned_Price[-rows.index]
head(train.data)
```

```
##        Id                                                Model Price Age_08_04
## 393   395 TOYOTA Corolla 1.6 16V VVT I WAGON TERRA Stationwagen  9950        49
## 1354 1360            TOYOTA Corolla 1.6 LB LINEA LUNA 4/5-Doors  8250        80
## 300   301      TOYOTA Corolla 1.6 16V VVT I HATCHB G6 2/3-Doors 13750        39
## 1404 1410               TOYOTA Corolla 1.3 16V HATCHB 2/3-Doors  7000        73
## 1339 1345  TOYOTA Corolla Hatchback 1.6 Terra Comfort 2/3-Doors  7499        80
## 721   724    TOYOTA Corolla 1.6 16V LIFTB LINEA TERRA 4/5-Doors  8450        63
##      Mfg_Month Mfg_Year     KM Fuel_Type  HP Met_Color Color Automatic   CC
## 393          8     2000 131364    Petrol 110         1 Black         0 1600
## 1354         1     1998  60476    Petrol 110         1  Grey         0 1600
## 300          6     2001  40000    Petrol 110         1  Grey         0 1600
## 1404         8     1998  47360    Petrol  86         0  Grey         0 1300
## 1339         1     1998  63500    Petrol 110         1  Blue         0 1600
## 721          6     1999  88685    Petrol 110         0  Grey         0 1600
##      Doors Cylinders Gears Quarterly_Tax Weight Mfr_Guarantee BOVAG_Guarantee
## 393      5         4     5            85   1075             0               1
## 1354     5         4     5            19   1114             0               1
## 300      3         4     5            85   1055             0               1
## 1404     3         4     5            69   1010             0               1
## 1339     3         4     5            69   1050             0               0
## 721      5         4     5            85   1070             1               1
##      Guarantee_Period ABS Airbag_1 Airbag_2 Airco Automatic_airco Boardcomputer
## 393                 3   1        1        1     0               0             1
## 1354                3   1        1        0     1               1             0
## 300                 3   1        1        1     1               0             1
## 1404                3   0        1        0     0               0             0
## 1339                3   0        1        0     0               0             0
## 721                 3   1        1        1     0               0             0
##      CD_Player Central_Lock Powered_Windows Power_Steering Radio Mistlamps
## 393          0            1               1              1     0         0
## 1354         0            1               1              1     0         1
```

```
## 300          0            1               1              1     0         1
## 1404         0            0               0              1     0         0
## 1339         0            1               1              1     0         0
## 721          0            0               0              1     0         0
##      Sport_Model Backseat_Divider Metallic_Rim Radio_cassette Parking_Assistant
## 393            0                1            0              0                 0
## 1354           0                0            0              0                 0
## 300            0                1            1              0                 0
## 1404           0                1            0              0                 0
## 1339           0                0            0              0                 0
## 721            1                1            0              0                 0
##      Tow_Bar Binned_Price
## 393        0            4
## 1354       0            3
## 300        1            7
## 1404       0            2
## 1339       1            3
## 721        0            3
```

```
RT.binned <- rpart(Binned_Price ~ Age_08_04 + KM + Fuel_Type +
HP + Automatic + Doors + Quarterly_Tax +
Mfr_Guarantee + Guarantee_Period + Airco +
Automatic_airco + CD_Player + Powered_Windows +
Sport_Model + Tow_Bar, data = train.data,minbucket=1)
prp(RT.binned)
```



Compare the tree generated by the CT with the one generated by the RT. Are they different? (Look at

structure, the top predictors, size of tree, etc.) Why? Answer: When creating bins the intent is to decrease the number of variables. This unsurpisingly results in the binned tree being significantly smaller than the original full tree. One interesting thing to note is the in our binned tree, CD Player replaces Quarterly Tax as one of the top four specifications in indicating price. This could be because tax variations are less significant within bins.

```
t(t(RT.binned$variable.importance))
```

```
##                         [,1]
## Age_08_04         140.372139
## KM                 72.439706
## CD_Player          25.003200
## Automatic_airco    19.178101
## Airco              18.794593
## Quarterly_Tax      17.627902
## Sport_Model        15.479258
## HP                  8.338291
## Powered_Windows     7.547571
## Fuel_Type           4.051562
## Mfr_Guarantee       3.774845
## Doors               3.273578
## Guarantee_Period    1.002471
```

Predict the price, using the RT and the CT, of a used Toyota Corolla with the specifications listed in Table 9.6. Compare the predictions in terms of the predictors that were used, the magnitude of the difference between the two predictions, and theadvantages and disadvantages of the two methods.

Answer:Our predictions for the two models were very simmilar. A difference of $32.78 (less than 1% of the total price of the car) is statistically insignificantin this case. Our binned model returned a whole number while the full model returned a more "accurate" price, but ultimately it is a wash. Both models had comparable accuracy, but the full regression seemed to be better trained. If we wanted to use the binned model I would suggestcreating smaller bin ranges to prevent underfitting the model. However, when considering the the overall accuracy range and the car sale market both models would be considered good enough for most used car sales markets.

```
#first create your new record
new.record <- data.frame(Age_08_04 = 77,
KM = 117000,
Fuel_Type = "Petrol",
HP = 110,
Automatic = 0,
Doors = 5,
Quarterly_Tax = 100,
Mfr_Guarantee = 0,
Guarantee_Period = 3,
Airco = 1,
Automatic_airco = 0,
CD_Player = 0,
Powered_Windows = 0,
Sport_Model = 0,
Tow_Bar = 1)
#set up your regression and classification trees
price.RT<- predict(RT, newdata = new.record)
#remember that we have bins for our CT
```

```r
price.RT.bin <- bins[predict(RT.binned, newdata = new.record, type = "class")]
cat(paste("Regression Price Estimate: ",scales::dollar(price.RT,0.01)),
paste("Classification Price Estimate: ",scales::dollar(price.RT.bin,0.01)),
sep='\n')
```

```
## Regression Price Estimate:  $7,132.22
## Classification Price Estimate:  $7,165.00
```