Assignment #2 Data Mining

2022-04-01

##Question 1: What is the key idea behind bagging? Can bagging deal both with high variance (overfitting) and high bias (under fitting)?

#The key idea behind bagging is to reduce the variance for algorithms that have high variance. Multiple weak learners can work better than a single strong learner. Bagging can only deal with high variance by reducing the variance. Bagging does not deal with the underfiting (the bias).

##Question 2: Why bagging models are computationally more efficient when compared to boosting models with the same number of weak learners?

#Bagging models are computationally more efficient compared to boosting models with the same number of weak learners because the base learners grow independently of each other in parallel, which reduces the overall computational complexity of the training phase.

##Question 3: James is thinking of creating an ensemble mode to predict whether a given stock will go up or down in the next week. He has trained several decision tree models but each model is not performing any better than a random model. The models are also very similar to each other. Do you think creating an ensemble model by combining these tree models can boost the performance?

#Creating an ensemble model by combining the tree models will not be able to boost performance because the models are not diverse. Since the models are very similar to each other, the performance will not be boosted. The key requirement of the ensemble models is to have a sufficiently diverse group of base learners. There is no point of combining similar base learners because they will have similar predictions.

##Question 4:Consider the following Table that classifies some objects into two classes of edible (+) and non- edible (-), based on some characteristics such as the object color, size and shape. What would be the Information gain for splitting the dataset based on the “Size” attribute?

#The information gain for splitting the data set based on the "size" attribute is .10578144.   
#This is calculated by using the information gain formula (info gain = entropy(parent) - [average entropy(children)])).   
#Information gain tells us how important a given attribute of the feature vector. So in this case, the information gain foe the size attribute is .10578144 important.

##Question 5: Why is it important that the m parameter (number of attributes available at each split) to be optimally set in random forest models? Discuss the implications of setting this parameter too small or too large.

#It is important that the m parameter to be optimally set in random forest models. This is because we want to achieve the most information gain and diversity in the data from nodes. This is so the first decision tree from beginning to end will have maximum diversity in the nodes. If the m parameter is set too small or too large, the model can be impure. There would not be enough information gain and diversity within the data.

#loading packages   
library(ISLR)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-3

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

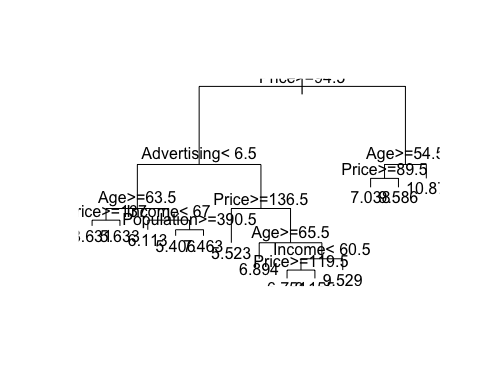
#Using dplyr to select sales, price, advertising, population, age, income, and education   
Carseats\_Filtered <- Carseats %>% select("Sales", "Price",   
"Advertising","Population","Age","Income","Education")

#Build a decision tree regression model to predict Sales based on all other attributes (“Price”, “Advertising”, “Population”, “Age”, “Income” and “Education”). Which attribute is used at the top of the tree (the root node) for splitting?Hint: you can either plot () and text()functions or use the summary() function to see the decision tree rules.

#install and loading packages needed  
#install.packages("rpart")  
#install.packages("rpart.plot")  
library(rpart)  
library(rpart.plot)  
mydata <- Carseats\_Filtered  
Model\_1 = rpart(Sales~.,data=mydata, method='anova')  
#Summary of Model 1  
summary(Model\_1)

## Call:  
## rpart(formula = Sales ~ ., data = mydata, method = "anova")  
## n= 400   
##   
## CP nsplit rel error xerror xstd  
## 1 0.14251535 0 1.0000000 1.0022970 0.06908019  
## 2 0.08034146 1 0.8574847 0.8983968 0.06428010  
## 3 0.06251702 2 0.7771432 0.8813326 0.06529801  
## 4 0.02925241 3 0.7146262 0.8019824 0.05705361  
## 5 0.02537341 4 0.6853738 0.8171027 0.05732470  
## 6 0.02127094 5 0.6600003 0.8096155 0.05554352  
## 7 0.02059174 6 0.6387294 0.7949710 0.05458821  
## 8 0.01632010 7 0.6181377 0.7839302 0.05429375  
## 9 0.01521801 8 0.6018176 0.7800797 0.05327425  
## 10 0.01042023 9 0.5865996 0.7630754 0.05207004  
## 11 0.01000559 10 0.5761793 0.7703483 0.05191005  
## 12 0.01000000 12 0.5561681 0.7703483 0.05191005  
##   
## Variable importance  
## Price Advertising Age Income Population Education   
## 49 18 16 8 6 3   
##   
## Node number 1: 400 observations, complexity param=0.1425153  
## mean=7.496325, MSE=7.955687   
## left son=2 (329 obs) right son=3 (71 obs)  
## Primary splits:  
## Price < 94.5 to the right, improve=0.14251530, (0 missing)  
## Advertising < 7.5 to the left, improve=0.07303226, (0 missing)  
## Age < 61.5 to the right, improve=0.07120203, (0 missing)  
## Income < 61.5 to the left, improve=0.02840494, (0 missing)  
## Population < 174.5 to the left, improve=0.01077467, (0 missing)  
##   
## Node number 2: 329 observations, complexity param=0.08034146  
## mean=7.001672, MSE=6.815199   
## left son=4 (174 obs) right son=5 (155 obs)  
## Primary splits:  
## Advertising < 6.5 to the left, improve=0.11402580, (0 missing)  
## Price < 136.5 to the right, improve=0.08411056, (0 missing)  
## Age < 63.5 to the right, improve=0.08091745, (0 missing)  
## Income < 60.5 to the left, improve=0.03394126, (0 missing)  
## Population < 23 to the left, improve=0.01831455, (0 missing)  
## Surrogate splits:  
## Population < 223 to the left, agree=0.599, adj=0.148, (0 split)  
## Education < 10.5 to the right, agree=0.565, adj=0.077, (0 split)  
## Age < 53.5 to the right, agree=0.547, adj=0.039, (0 split)  
## Income < 114.5 to the left, agree=0.547, adj=0.039, (0 split)  
## Price < 106.5 to the right, agree=0.544, adj=0.032, (0 split)  
##   
## Node number 3: 71 observations, complexity param=0.02537341  
## mean=9.788451, MSE=6.852836   
## left son=6 (36 obs) right son=7 (35 obs)  
## Primary splits:  
## Age < 54.5 to the right, improve=0.16595410, (0 missing)  
## Price < 75.5 to the right, improve=0.08365773, (0 missing)  
## Income < 30.5 to the left, improve=0.03322169, (0 missing)  
## Education < 10.5 to the right, improve=0.03019634, (0 missing)  
## Population < 268.5 to the left, improve=0.02383306, (0 missing)  
## Surrogate splits:  
## Advertising < 4.5 to the right, agree=0.606, adj=0.200, (0 split)  
## Price < 73 to the right, agree=0.592, adj=0.171, (0 split)  
## Population < 272.5 to the left, agree=0.592, adj=0.171, (0 split)  
## Income < 79.5 to the right, agree=0.592, adj=0.171, (0 split)  
## Education < 11.5 to the left, agree=0.577, adj=0.143, (0 split)  
##   
## Node number 4: 174 observations, complexity param=0.02127094  
## mean=6.169655, MSE=4.942347   
## left son=8 (58 obs) right son=9 (116 obs)  
## Primary splits:  
## Age < 63.5 to the right, improve=0.078712160, (0 missing)  
## Price < 130.5 to the right, improve=0.048919280, (0 missing)  
## Population < 26.5 to the left, improve=0.030421540, (0 missing)  
## Income < 67.5 to the left, improve=0.027749670, (0 missing)  
## Advertising < 0.5 to the left, improve=0.006795377, (0 missing)  
## Surrogate splits:  
## Income < 22.5 to the left, agree=0.678, adj=0.034, (0 split)  
## Price < 96.5 to the left, agree=0.672, adj=0.017, (0 split)  
## Population < 26.5 to the left, agree=0.672, adj=0.017, (0 split)  
##   
## Node number 5: 155 observations, complexity param=0.06251702  
## mean=7.935677, MSE=7.268151   
## left son=10 (28 obs) right son=11 (127 obs)  
## Primary splits:  
## Price < 136.5 to the right, improve=0.17659580, (0 missing)  
## Age < 73.5 to the right, improve=0.08000201, (0 missing)  
## Income < 60.5 to the left, improve=0.05360755, (0 missing)  
## Advertising < 13.5 to the left, improve=0.03920507, (0 missing)  
## Population < 399 to the left, improve=0.01037956, (0 missing)  
## Surrogate splits:  
## Advertising < 24.5 to the right, agree=0.826, adj=0.036, (0 split)  
##   
## Node number 6: 36 observations, complexity param=0.0163201  
## mean=8.736944, MSE=4.961043   
## left son=12 (12 obs) right son=13 (24 obs)  
## Primary splits:  
## Price < 89.5 to the right, improve=0.29079360, (0 missing)  
## Income < 39.5 to the left, improve=0.19043350, (0 missing)  
## Advertising < 11.5 to the left, improve=0.17891930, (0 missing)  
## Age < 75.5 to the right, improve=0.04316067, (0 missing)  
## Education < 14.5 to the left, improve=0.03411396, (0 missing)  
## Surrogate splits:  
## Advertising < 16.5 to the right, agree=0.722, adj=0.167, (0 split)  
## Income < 37.5 to the left, agree=0.722, adj=0.167, (0 split)  
## Age < 56.5 to the left, agree=0.694, adj=0.083, (0 split)  
##   
## Node number 7: 35 observations  
## mean=10.87, MSE=6.491674   
##   
## Node number 8: 58 observations, complexity param=0.01042023  
## mean=5.287586, MSE=3.93708   
## left son=16 (10 obs) right son=17 (48 obs)  
## Primary splits:  
## Price < 137 to the right, improve=0.14521540, (0 missing)  
## Education < 15.5 to the right, improve=0.07995394, (0 missing)  
## Income < 35.5 to the left, improve=0.04206708, (0 missing)  
## Age < 79.5 to the left, improve=0.02799057, (0 missing)  
## Population < 52.5 to the left, improve=0.01914342, (0 missing)  
##   
## Node number 9: 116 observations, complexity param=0.01000559  
## mean=6.61069, MSE=4.861446   
## left son=18 (58 obs) right son=19 (58 obs)  
## Primary splits:  
## Income < 67 to the left, improve=0.05085914, (0 missing)  
## Population < 392 to the right, improve=0.04476721, (0 missing)  
## Price < 127 to the right, improve=0.04210762, (0 missing)  
## Age < 37.5 to the right, improve=0.02858424, (0 missing)  
## Education < 14.5 to the left, improve=0.01187387, (0 missing)  
## Surrogate splits:  
## Education < 12.5 to the right, agree=0.586, adj=0.172, (0 split)  
## Age < 58.5 to the left, agree=0.578, adj=0.155, (0 split)  
## Price < 144.5 to the left, agree=0.569, adj=0.138, (0 split)  
## Population < 479 to the right, agree=0.560, adj=0.121, (0 split)  
## Advertising < 2.5 to the right, agree=0.543, adj=0.086, (0 split)  
##   
## Node number 10: 28 observations  
## mean=5.522857, MSE=5.084213   
##   
## Node number 11: 127 observations, complexity param=0.02925241  
## mean=8.467638, MSE=6.183142   
## left son=22 (29 obs) right son=23 (98 obs)  
## Primary splits:  
## Age < 65.5 to the right, improve=0.11854590, (0 missing)  
## Income < 51.5 to the left, improve=0.08076060, (0 missing)  
## Advertising < 13.5 to the left, improve=0.04801701, (0 missing)  
## Education < 11.5 to the right, improve=0.02471512, (0 missing)  
## Population < 479 to the left, improve=0.01908657, (0 missing)  
##   
## Node number 12: 12 observations  
## mean=7.038333, MSE=2.886964   
##   
## Node number 13: 24 observations  
## mean=9.58625, MSE=3.834123   
##   
## Node number 16: 10 observations  
## mean=3.631, MSE=5.690169   
##   
## Node number 17: 48 observations  
## mean=5.632708, MSE=2.88102   
##   
## Node number 18: 58 observations  
## mean=6.113448, MSE=3.739109   
##   
## Node number 19: 58 observations, complexity param=0.01000559  
## mean=7.107931, MSE=5.489285   
## left son=38 (10 obs) right son=39 (48 obs)  
## Primary splits:  
## Population < 390.5 to the right, improve=0.10993270, (0 missing)  
## Price < 124.5 to the right, improve=0.07534567, (0 missing)  
## Advertising < 0.5 to the left, improve=0.07060488, (0 missing)  
## Age < 45.5 to the right, improve=0.04611510, (0 missing)  
## Education < 11.5 to the right, improve=0.03722944, (0 missing)  
##   
## Node number 22: 29 observations  
## mean=6.893793, MSE=6.08343   
##   
## Node number 23: 98 observations, complexity param=0.02059174  
## mean=8.933367, MSE=5.262759   
## left son=46 (34 obs) right son=47 (64 obs)  
## Primary splits:  
## Income < 60.5 to the left, improve=0.12705480, (0 missing)  
## Advertising < 13.5 to the left, improve=0.07114001, (0 missing)  
## Price < 118.5 to the right, improve=0.06932216, (0 missing)  
## Education < 11.5 to the right, improve=0.03377416, (0 missing)  
## Age < 49.5 to the right, improve=0.02289004, (0 missing)  
## Surrogate splits:  
## Education < 17.5 to the right, agree=0.663, adj=0.029, (0 split)  
##   
## Node number 38: 10 observations  
## mean=5.406, MSE=2.508524   
##   
## Node number 39: 48 observations  
## mean=7.4625, MSE=5.381106   
##   
## Node number 46: 34 observations, complexity param=0.01521801  
## mean=7.811471, MSE=4.756548   
## left son=92 (19 obs) right son=93 (15 obs)  
## Primary splits:  
## Price < 119.5 to the right, improve=0.29945020, (0 missing)  
## Advertising < 11.5 to the left, improve=0.14268440, (0 missing)  
## Income < 40.5 to the right, improve=0.12781140, (0 missing)  
## Population < 152 to the left, improve=0.03601768, (0 missing)  
## Age < 49.5 to the right, improve=0.02748814, (0 missing)  
## Surrogate splits:  
## Education < 12.5 to the right, agree=0.676, adj=0.267, (0 split)  
## Advertising < 7.5 to the right, agree=0.647, adj=0.200, (0 split)  
## Age < 53.5 to the left, agree=0.647, adj=0.200, (0 split)  
## Population < 240 to the right, agree=0.618, adj=0.133, (0 split)  
## Income < 41.5 to the right, agree=0.618, adj=0.133, (0 split)  
##   
## Node number 47: 64 observations  
## mean=9.529375, MSE=4.5078   
##   
## Node number 92: 19 observations  
## mean=6.751053, MSE=3.378915   
##   
## Node number 93: 15 observations  
## mean=9.154667, MSE=3.273025

#Plotting model 1   
plot(Model\_1)  
text(Model\_1)



#The attribute that is at the top of the tree is Price.

#Question #2: Consider the following input:Sales=9, Price=6.54, Population=124, Advertising=0, Age=76, Income= 110, Education=10. What will be the estimated Sales for this record using the decision tree model?

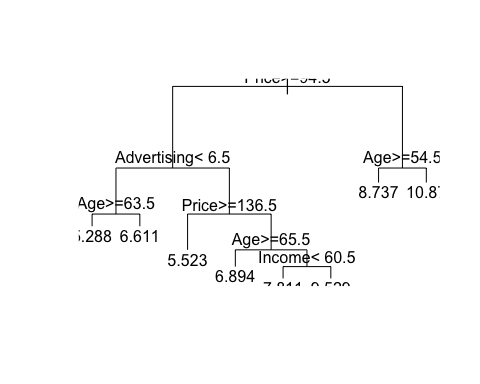
Model\_2 = rpart(Sales~.,data=mydata, method='anova', control = rpart.control(minsplit = 60 ))  
summary(Model\_2)

## Call:  
## rpart(formula = Sales ~ ., data = mydata, method = "anova", control = rpart.control(minsplit = 60))  
## n= 400   
##   
## CP nsplit rel error xerror xstd  
## 1 0.14251535 0 1.0000000 1.0022563 0.06921614  
## 2 0.08034146 1 0.8574847 0.9186463 0.06534075  
## 3 0.06251702 2 0.7771432 0.8943072 0.06653013  
## 4 0.02925241 3 0.7146262 0.8086993 0.05894532  
## 5 0.02537341 4 0.6853738 0.8053006 0.05733500  
## 6 0.02127094 5 0.6600003 0.7814354 0.05590569  
## 7 0.02059174 6 0.6387294 0.7625923 0.05409808  
## 8 0.01000000 7 0.6181377 0.7272948 0.05222955  
##   
## Variable importance  
## Price Advertising Age Income Population Education   
## 49 20 18 7 4 2   
##   
## Node number 1: 400 observations, complexity param=0.1425153  
## mean=7.496325, MSE=7.955687   
## left son=2 (329 obs) right son=3 (71 obs)  
## Primary splits:  
## Price < 94.5 to the right, improve=0.14251530, (0 missing)  
## Advertising < 7.5 to the left, improve=0.07303226, (0 missing)  
## Age < 61.5 to the right, improve=0.07120203, (0 missing)  
## Income < 61.5 to the left, improve=0.02840494, (0 missing)  
## Population < 174.5 to the left, improve=0.01077467, (0 missing)  
##   
## Node number 2: 329 observations, complexity param=0.08034146  
## mean=7.001672, MSE=6.815199   
## left son=4 (174 obs) right son=5 (155 obs)  
## Primary splits:  
## Advertising < 6.5 to the left, improve=0.11402580, (0 missing)  
## Price < 136.5 to the right, improve=0.08411056, (0 missing)  
## Age < 63.5 to the right, improve=0.08091745, (0 missing)  
## Income < 60.5 to the left, improve=0.03394126, (0 missing)  
## Population < 174.5 to the left, improve=0.01484238, (0 missing)  
## Surrogate splits:  
## Population < 223 to the left, agree=0.599, adj=0.148, (0 split)  
## Education < 10.5 to the right, agree=0.565, adj=0.077, (0 split)  
## Age < 53.5 to the right, agree=0.547, adj=0.039, (0 split)  
## Income < 114.5 to the left, agree=0.547, adj=0.039, (0 split)  
## Price < 106.5 to the right, agree=0.544, adj=0.032, (0 split)  
##   
## Node number 3: 71 observations, complexity param=0.02537341  
## mean=9.788451, MSE=6.852836   
## left son=6 (36 obs) right son=7 (35 obs)  
## Primary splits:  
## Age < 54.5 to the right, improve=0.16595410, (0 missing)  
## Price < 77.5 to the right, improve=0.08080275, (0 missing)  
## Population < 268.5 to the left, improve=0.02383306, (0 missing)  
## Income < 57 to the left, improve=0.02353594, (0 missing)  
## Education < 12.5 to the right, improve=0.02237407, (0 missing)  
## Surrogate splits:  
## Advertising < 4.5 to the right, agree=0.606, adj=0.200, (0 split)  
## Price < 73 to the right, agree=0.592, adj=0.171, (0 split)  
## Population < 272.5 to the left, agree=0.592, adj=0.171, (0 split)  
## Income < 79.5 to the right, agree=0.592, adj=0.171, (0 split)  
## Education < 11.5 to the left, agree=0.577, adj=0.143, (0 split)  
##   
## Node number 4: 174 observations, complexity param=0.02127094  
## mean=6.169655, MSE=4.942347   
## left son=8 (58 obs) right son=9 (116 obs)  
## Primary splits:  
## Age < 63.5 to the right, improve=0.078712160, (0 missing)  
## Price < 130.5 to the right, improve=0.048919280, (0 missing)  
## Income < 67.5 to the left, improve=0.027749670, (0 missing)  
## Population < 326 to the right, improve=0.020525710, (0 missing)  
## Advertising < 0.5 to the left, improve=0.006795377, (0 missing)  
## Surrogate splits:  
## Income < 22.5 to the left, agree=0.678, adj=0.034, (0 split)  
## Price < 96.5 to the left, agree=0.672, adj=0.017, (0 split)  
## Population < 26.5 to the left, agree=0.672, adj=0.017, (0 split)  
##   
## Node number 5: 155 observations, complexity param=0.06251702  
## mean=7.935677, MSE=7.268151   
## left son=10 (28 obs) right son=11 (127 obs)  
## Primary splits:  
## Price < 136.5 to the right, improve=0.17659580, (0 missing)  
## Age < 65.5 to the right, improve=0.07915291, (0 missing)  
## Income < 60.5 to the left, improve=0.05360755, (0 missing)  
## Advertising < 13.5 to the left, improve=0.03920507, (0 missing)  
## Population < 399 to the left, improve=0.01037956, (0 missing)  
## Surrogate splits:  
## Advertising < 24.5 to the right, agree=0.826, adj=0.036, (0 split)  
##   
## Node number 6: 36 observations  
## mean=8.736944, MSE=4.961043   
##   
## Node number 7: 35 observations  
## mean=10.87, MSE=6.491674   
##   
## Node number 8: 58 observations  
## mean=5.287586, MSE=3.93708   
##   
## Node number 9: 116 observations  
## mean=6.61069, MSE=4.861446   
##   
## Node number 10: 28 observations  
## mean=5.522857, MSE=5.084213   
##   
## Node number 11: 127 observations, complexity param=0.02925241  
## mean=8.467638, MSE=6.183142   
## left son=22 (29 obs) right son=23 (98 obs)  
## Primary splits:  
## Age < 65.5 to the right, improve=0.11854590, (0 missing)  
## Income < 51.5 to the left, improve=0.08076060, (0 missing)  
## Advertising < 13.5 to the left, improve=0.04801701, (0 missing)  
## Education < 11.5 to the right, improve=0.02471512, (0 missing)  
## Population < 405 to the left, improve=0.01719030, (0 missing)  
##   
## Node number 22: 29 observations  
## mean=6.893793, MSE=6.08343   
##   
## Node number 23: 98 observations, complexity param=0.02059174  
## mean=8.933367, MSE=5.262759   
## left son=46 (34 obs) right son=47 (64 obs)  
## Primary splits:  
## Income < 60.5 to the left, improve=0.12705480, (0 missing)  
## Advertising < 13.5 to the left, improve=0.07114001, (0 missing)  
## Price < 118.5 to the right, improve=0.06932216, (0 missing)  
## Education < 11.5 to the right, improve=0.03377416, (0 missing)  
## Age < 49.5 to the right, improve=0.02289004, (0 missing)  
## Surrogate splits:  
## Education < 17.5 to the right, agree=0.663, adj=0.029, (0 split)  
##   
## Node number 46: 34 observations  
## mean=7.811471, MSE=4.756548   
##   
## Node number 47: 64 observations  
## mean=9.529375, MSE=4.5078

print(Model\_2)

## n= 400   
##   
## node), split, n, deviance, yval  
## \* denotes terminal node  
##   
## 1) root 400 3182.2750 7.496325   
## 2) Price>=94.5 329 2242.2000 7.001672   
## 4) Advertising< 6.5 174 859.9684 6.169655   
## 8) Age>=63.5 58 228.3507 5.287586 \*  
## 9) Age< 63.5 116 563.9277 6.610690 \*  
## 5) Advertising>=6.5 155 1126.5630 7.935677   
## 10) Price>=136.5 28 142.3580 5.522857 \*  
## 11) Price< 136.5 127 785.2591 8.467638   
## 22) Age>=65.5 29 176.4195 6.893793 \*  
## 23) Age< 65.5 98 515.7504 8.933367   
## 46) Income< 60.5 34 161.7226 7.811471 \*  
## 47) Income>=60.5 64 288.4992 9.529375 \*  
## 3) Price< 94.5 71 486.5513 9.788451   
## 6) Age>=54.5 36 178.5976 8.736944 \*  
## 7) Age< 54.5 35 227.2086 10.870000 \*

plot(Model\_2)  
text(Model\_2)



new\_model <- data.frame(Price=6.54, Population=124, Advertising=0, Age=76, Income= 110, Education=10)  
predict(Model\_1, newdata=new\_model)

## 1   
## 9.58625

#The estimated sales for this record using a decision tree model is 9.5862.

#Question 3: Use the caret function to train a random forest (method=’rf’) for the same dataset. Use the caret default settings. By default, caret will examine the “mtry” values of 2,4, and 6. Recall that mtry is the number of attributes available for splitting at each splitting node. Which mtry value gives the best performance?   
size = floor(0.70\*nrow(Carseats\_Filtered))  
size

## [1] 280

set.seed(123)  
train = sample(seq\_len(nrow(Carseats\_Filtered)), size = size)  
traindata = Carseats\_Filtered[train,]  
testdata = Carseats\_Filtered[-train,]  
rf\_tree <- train(Sales~.,  
 data = Carseats\_Filtered,  
 method = "rf")  
print(rf\_tree)

## Random Forest   
##   
## 400 samples  
## 6 predictor  
##   
## No pre-processing  
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 400, 400, 400, 400, 400, 400, ...   
## Resampling results across tuning parameters:  
##   
## mtry RMSE Rsquared MAE   
## 2 2.432085 0.2777412 1.957961  
## 4 2.449238 0.2728587 1.965729  
## 6 2.474999 0.2626543 1.985643  
##   
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was mtry = 2.

#The mtry that gives the best performance is the 2nd mtry.

#Question 4: Customize the search grid by checking the model’s performance for mtry values of 2, 3 and 5 using 3 repeats of 5-fold cross validation

#Using mtry value of 2  
mtry = 2  
train\_2 <- trainControl(method = "repeatedcv", number = 5, repeats = 3)  
tunegrid1 <- expand.grid(.mtry=mtry)  
tree2 <- train(Sales~.,  
 method = "rf",  
 data = traindata,  
 trControl = train\_2,  
 tuneGrid=tunegrid1  
 )  
print(tree2)

## Random Forest   
##   
## 280 samples  
## 6 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold, repeated 3 times)   
## Summary of sample sizes: 224, 224, 224, 224, 224, 224, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 2.433514 0.2623944 1.951287  
##   
## Tuning parameter 'mtry' was held constant at a value of 2

#Using mtry value of 3  
mtry = 3  
train\_2 <- trainControl(method = "repeatedcv", number = 5, repeats = 3)  
tunegrid <- expand.grid(.mtry=mtry)  
tree2 <- train(Sales~.,  
 method = "rf",  
 data = traindata,  
 trControl = train\_2,  
 tuneGrid=tunegrid  
 )  
print(tree2)

## Random Forest   
##   
## 280 samples  
## 6 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold, repeated 3 times)   
## Summary of sample sizes: 224, 224, 224, 224, 224, 224, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 2.411611 0.2755602 1.927487  
##   
## Tuning parameter 'mtry' was held constant at a value of 3

#Using mtry value of 5  
mtry = 5  
train\_2 <- trainControl(method = "repeatedcv", number = 5, repeats = 3)  
tunegrid <- expand.grid(.mtry=mtry)  
tree2 <- train(Sales~.,  
 method = "rf",  
 data = traindata,  
 trControl = train\_2,  
 tuneGrid=tunegrid  
 )  
print(tree2)

## Random Forest   
##   
## 280 samples  
## 6 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold, repeated 3 times)   
## Summary of sample sizes: 224, 224, 224, 224, 224, 224, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 2.431127 0.2711105 1.963302  
##   
## Tuning parameter 'mtry' was held constant at a value of 5