Week6_lab

Zaiwei
01/09/2019

Start R studio

Create a New Project

Install Package "ISLR" (If not installed earlier)

Upload following data sets "Carseats" ,

View the data sets

```
library(ISLR)
library(tree)
attach(Carseats)
dim(Carseats)
## [1] 400  11
names(Carseats)
##  [1] "Sales"       "CompPrice"   "Income"      "Advertising" "Population"
##  [6] "Price"       "ShelveLoc"   "Age"         "Education"   "Urban"
## [11] "US"
sapply(Carseats, class)
##       Sales   CompPrice      Income Advertising  Population       Price
##   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"
##   ShelveLoc         Age   Education       Urban          US
##    "factor"   "numeric"   "numeric"    "factor"    "factor"
```

Decision Trees: Use "Carseats" data set with Sales as the Target Variable

Fit a Regression Tree for training data set

Construct the Cross validation plot

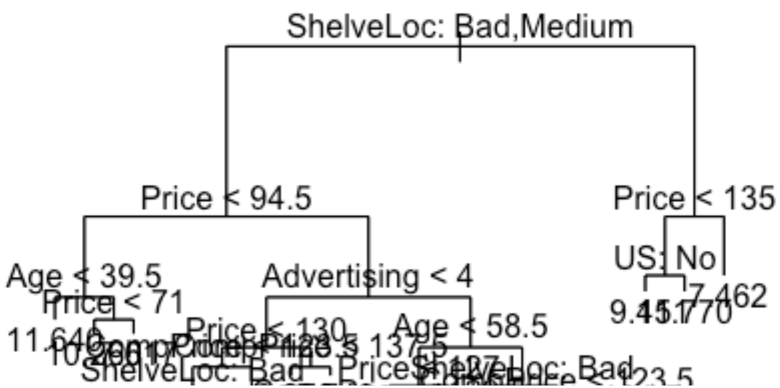
Select the best size of the tree

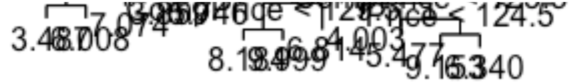
Obtain the best regression tree by pruning

Test the model accuracy

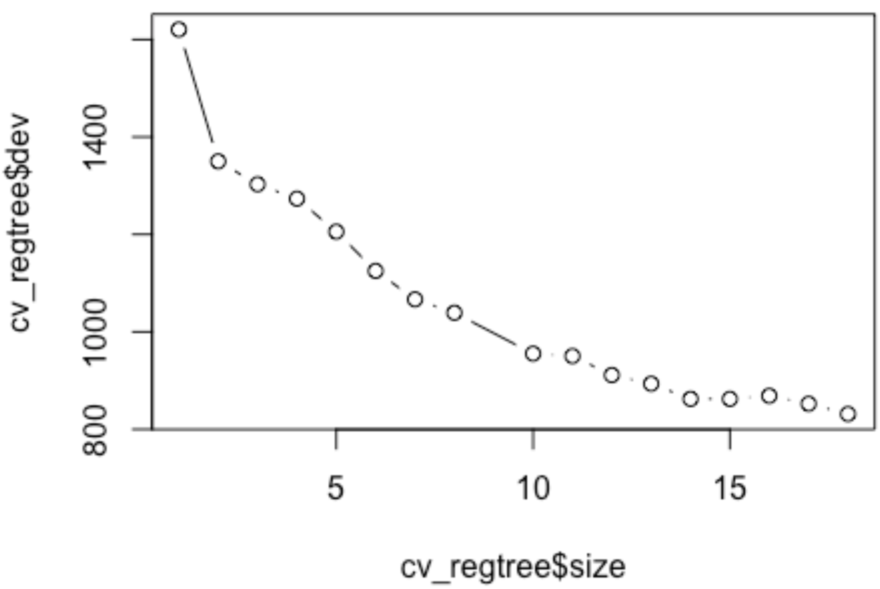
Describe the terminal nodes of the resulting decision tree

```
set.seed(1)
trainData<-sample(1:nrow(Carseats),200)
reg_tree<-tree(Sales~., Carseats,subset = trainData)
plot(reg_tree)
text(reg_tree, pretty = 0)
```



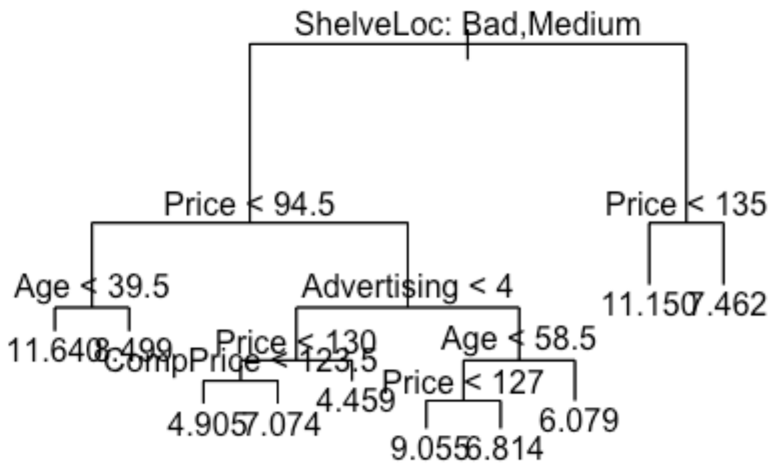


```
summary(reg_tree)
##
## Regression tree:
## tree(formula = Sales ~ ., data = Carseats, subset = trainData)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Age"         "Advertising" "CompPrice"
## [6] "US"
## Number of terminal nodes:  18
## Residual mean deviance:  2.167 = 394.3 / 182
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -3.88200 -0.88200 -0.08712  0.00000  0.89590  4.09900
cv_regtree <- cv.tree(reg_tree)
cv_regtree
## $size
##  [1] 18 17 16 15 14 13 12 11 10  8  7  6  5  4  3  2  1
##
## $dev
##  [1]  831.3437  852.3639  868.6815  862.3400  862.3400  893.4641  911.2580
##  [8]  950.2691  955.2535 1039.1241 1066.6899 1125.0894 1205.5806 1273.2889
## [15] 1302.8607 1349.9273 1620.4687
##
## $k
##  [1]      -Inf  16.99544  20.56322  25.01730  25.57104  28.01938  30.36962
##  [8]  31.56747  31.80816  40.75445  44.44673  52.57126  76.21881  99.59459
## [15] 116.69889 159.79501 337.60153
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
plot(cv_regtree$size,cv_regtree$dev,type="b")
```




```
pruned_regtree <- prune.tree(reg_tree,best=10)
pruned_regtree
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 200 1573.000  7.578
```

```
##   2) ShelveLoc: Bad,Medium 158  964.600  6.908
##     4) Price < 94.5 24  110.900  9.285
##       8) Age < 39.5 6    9.117 11.640 *
##       9) Age > 39.5 18   57.310  8.499 *
##     5) Price > 94.5 134  694.000  6.483
##      10) Advertising < 4 59  229.900  5.511
##        20) Price < 130 37  135.300  6.136
##          40) CompPrice < 123.5 16   45.860  4.905 *
##          41) CompPrice > 123.5 21   46.760  7.074 *
##        21) Price > 130 22   55.810  4.459 *
##      11) Advertising > 4 75  364.400  7.247
##        22) Age < 58.5 43  138.700  8.117
##          44) Price < 127 25   55.660  9.055 *
##          45) Price > 127 18   30.440  6.814 *
##        23) Age > 58.5 32  149.500  6.079 *
##   3) ShelveLoc: Good 42  270.500 10.100
##     6) Price < 135 30  116.900 11.150 *
##     7) Price > 135 12   36.890  7.462 *
plot(pruned_regtree)
text(pruned_regtree,pretty=0)
```




```
yhat <- predict(pruned_regtree,newdata = Carseats[-trainData])
Carseats_test <- Carseats[-trainData,"Sales"]
MSE <- mean((yhat-Carseats_test)^2)
MSE
## [1] 12.82276
```

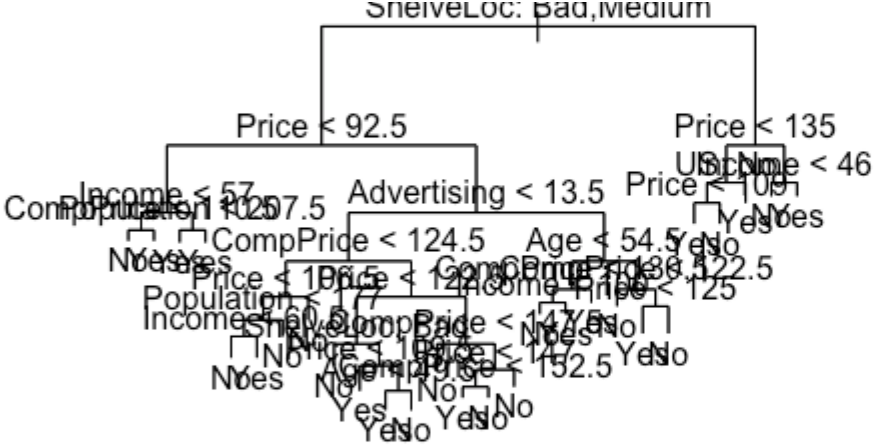
Decision Trees: Use “Carseats” data set with Sales as the Target Variable

Transfer Sales Variable from a Continuous variable to a Categorical Variable

```
High<-ifelse(Sales<=8,"No","Yes")
Carseats_new<-data.frame(Carseats,High)
Carseats_New<-Carseats_new[,-1]
```

3.2 Fit a Classification Tree for the full data set

```
tree_carseats<-tree(High~.,Carseats_New)
plot(tree_carseats)
text(tree_carseats, pretty=0)
```



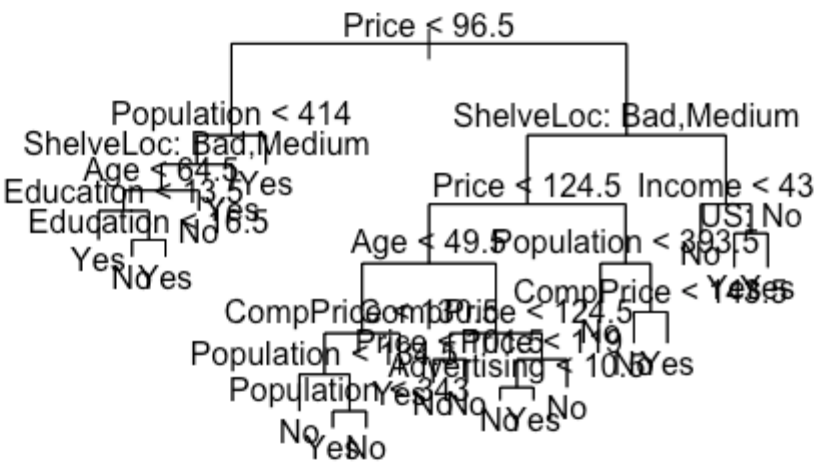
#

Test the model accuracy

```
summary(tree_carseats)
##
## Classification tree:
## tree(formula = High ~ ., data = Carseats_New)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Income"      "CompPrice"   "Population"
## [6] "Advertising" "Age"         "US"
## Number of terminal nodes:  27
## Residual mean deviance:  0.4575 = 170.7 / 373
## Misclassification error rate: 0.09 = 36 / 400
```

#3.4 Fit a Classification Tree for training data set

```
set.seed(2)
train<-sample(1:nrow(Carseats_New),200)
test<-Carseats_New[-train,]
tree_carsets_train<-tree(High~.,Carseats_New, subset = train)
plot(tree_carsets_train)
text(tree_carsets_train, pretty=0)
```



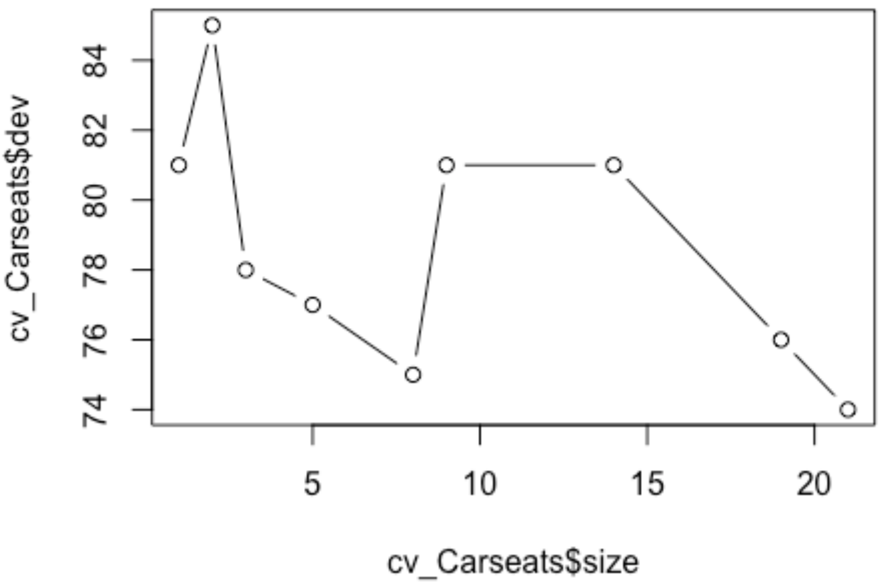
#3.5 Test the model accuracy

```
tree_predicted<-predict(tree_carsets_train, test, type="class")
High_test<-High[-train]
table(tree_predicted,High_test)
##               High_test
## tree_predicted  No Yes
##            No  104  33
```

```
##           Yes  13  50
matrix<-table(tree_predicted,High_test)
misrate<-((matrix[1,2]+matrix[2,1])/sum(matrix))
paste("Misclassification error rate is ",misrate)
## [1] "Misclassification error rate is  0.23"
```

3.6 Construct the Cross validation plot

```
set.seed(3)
cv_Carseats<-cv.tree(tree_carsets_train, FUN = prune.misclass)
names(cv_Carseats)
## [1] "size"   "dev"    "k"      "method"
cv_Carseats
## $size
## [1] 21 19 14  9  8  5  3  2  1
##
## $dev
## [1] 74 76 81 81 75 77 78 85 81
##
## $k
## [1] -Inf  0.0  1.0  1.4  2.0  3.0  4.0  9.0 18.0
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
plot(cv_Carseats$size, cv_Carseats$dev, type = "b")
```

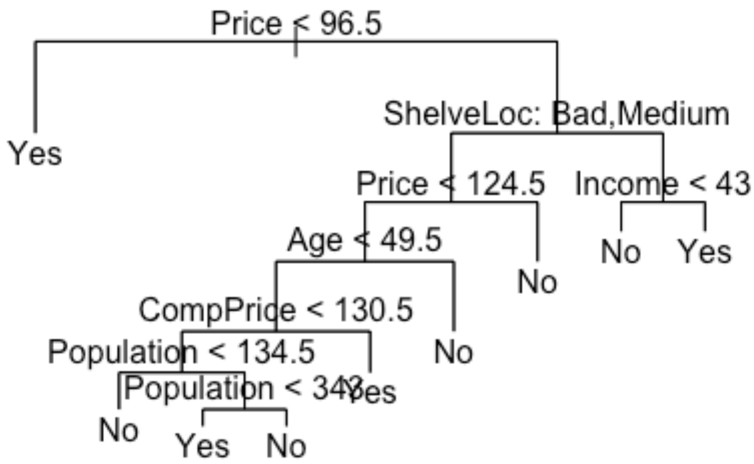


#

3.7 Select the best size of the tree model ### tree model is 9 # 3.8 Obtain the best classification tree by pruning

```
pruned_carseats<-prune.misclass(tree_carsets_train, best = 9)
pruned_carseats
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##   1) root 200 270.000 No ( 0.59500 0.40500 )
##     2) Price < 96.5 40  47.050 Yes ( 0.27500 0.72500 ) *
##     3) Price > 96.5 160 201.800 No ( 0.67500 0.32500 )
##       6) ShelveLoc: Bad,Medium 135 154.500 No ( 0.74074 0.25926 )
##        12) Price < 124.5 82 107.700 No ( 0.63415 0.36585 )
##          24) Age < 49.5 34  45.230 Yes ( 0.38235 0.61765 )
##            48) CompPrice < 130.5 21  28.680 No ( 0.57143 0.42857 )
##              96) Population < 134.5 6   0.000 No ( 1.00000 0.00000 ) *
##              97) Population > 134.5 15  20.190 Yes ( 0.40000 0.60000 )
##               194) Population < 343 7   5.742 Yes ( 0.14286 0.85714 ) *
##               195) Population > 343 8  10.590 No ( 0.62500 0.37500 ) *
##            49) CompPrice > 130.5 13   7.051 Yes ( 0.07692 0.92308 ) *
```

```
##         25) Age > 49.5 48  46.330 No ( 0.81250 0.18750 ) *
##       13) Price > 124.5 53  33.120 No ( 0.90566 0.09434 ) *
##     7) ShelveLoc: Good 25  31.340 Yes ( 0.32000 0.68000 )
##       14) Income < 43 7   8.376 No ( 0.71429 0.28571 ) *
##       15) Income > 43 18  16.220 Yes ( 0.16667 0.83333 ) *
plot(pruned_carseats)
text(pruned_carseats,pretty=0)
```



#

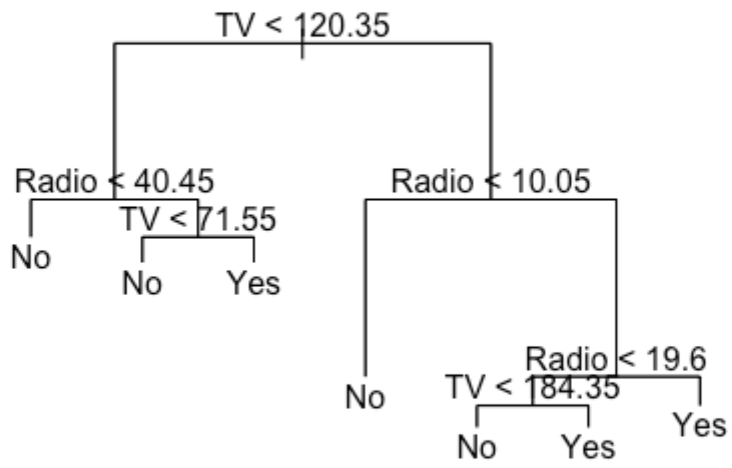
3.9 Test the model accuracy

```
tree_predict<-predict(pruned_carseats, test,type="class")
matrix2<-table(tree_predict,High_test)
mis_rate<-((matrix2[1,2]+matrix2[2,1])/sum(matrix2))
paste("Misclassification error rate is ",mis_rate)
## [1] "Misclassification error rate is  0.225"
```

4. Repeat above using Advertising data set if you have finished early and upload via Turnitin link in vUWS.

```
Advertising <- read.csv('Advertising.csv')
attach(Advertising)
## The following object is masked from Carseats:
##
##     Sales
names(Advertising)
## [1] "TV"        "Radio"     "Newspaper" "Sales"
sapply(Advertising,class)
##        TV     Radio Newspaper     Sales
## "numeric" "numeric" "numeric" "numeric"
High_Sales_adver <- ifelse(Sales<=14,"No","Yes")
High_Sales_adver
##   [1] "Yes" "No"  "No"  "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "No"
##  [12] "Yes" "No"  "No"  "Yes" "Yes" "No"  "Yes" "No"  "Yes" "Yes" "No"
##  [23] "No"  "Yes" "No"  "No"  "Yes" "Yes" "Yes" "No"  "Yes" "No"  "No"
##  [34] "Yes" "No"  "No"  "Yes" "Yes" "No"  "Yes" "Yes" "Yes" "Yes" "No"
##  [45] "No"  "Yes" "No"  "Yes" "Yes" "No"  "No"  "No"  "Yes" "Yes" "Yes"
##  [56] "Yes" "No"  "No"  "Yes" "Yes" "No"  "Yes" "Yes" "No"  "Yes" "No"
##  [67] "No"  "No"  "Yes" "Yes" "Yes" "No"  "No"  "No"  "Yes" "No"  "No"
##  [78] "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "Yes" "Yes" "No"  "Yes"
##  [89] "No"  "Yes" "No"  "No"  "Yes" "Yes" "No"  "Yes" "No"  "Yes" "Yes"
## [100] "Yes" "No"  "Yes" "Yes" "Yes" "Yes" "Yes" "No"  "No"  "No"  "Yes"
## [111] "No"  "Yes" "Yes" "Yes" "Yes" "No"  "No"  "No"  "Yes" "No"  "Yes"
## [122] "No"  "No"  "Yes" "Yes" "No"  "No"  "No"  "Yes" "No"  "No"  "No"
## [133] "No"  "Yes" "No"  "No"  "No"  "Yes" "No"  "Yes" "No"  "Yes" "Yes"
## [144] "No"  "No"  "No"  "No"  "Yes" "No"  "No"  "Yes" "No"  "Yes" "Yes"
## [155] "Yes" "No"  "Yes" "No"  "No"  "No"  "Yes" "No"  "Yes" "Yes" "No"
## [166] "No"  "No"  "No"  "Yes" "Yes" "No"  "Yes" "No"  "No"  "No"  "Yes"
## [177] "Yes" "No"  "No"  "No"  "No"  "No"  "No"  "Yes" "Yes" "Yes" "No"
## [188] "Yes" "Yes" "No"  "No"  "No"  "No"  "Yes" "Yes" "No"  "No"  "No"
```

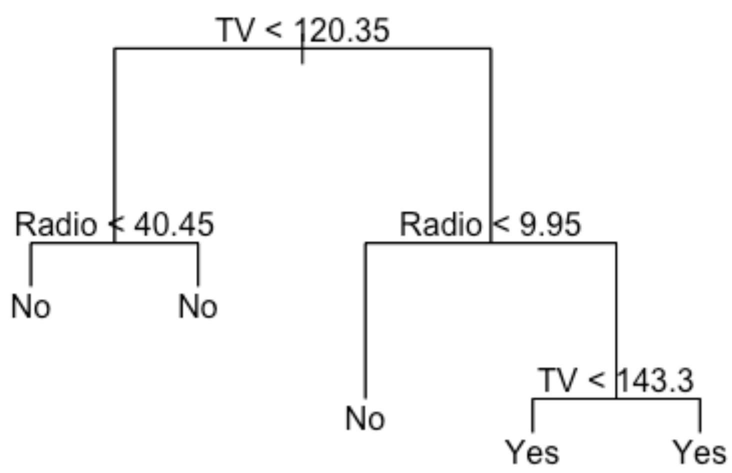
```
## [199] "Yes" "No"
Advertising_new <- data.frame(Advertising,High_Sales_adver)
Advertising_new <- Advertising_new[,-4]
tree_Advertising <- tree(High_Sales_adver~.,Advertising_new)
plot(tree_Advertising)
text(tree_Advertising,pretty=0)
```




```
summary(tree_Advertising)
##
## Classification tree:
## tree(formula = High_Sales_adver ~ ., data = Advertising_new)
## Variables actually used in tree construction:
## [1] "TV"    "Radio"
## Number of terminal nodes:  7
## Residual mean deviance:  0.0434 = 8.376 / 193
## Misclassification error rate: 0.01 = 2 / 200
```

Fit a Classification Tree for training data set

```
set.seed(2)
train_advert <- sample(1:nrow(Advertising_new),100)
test_advert <- Advertising_new[-train_advert,]
tree_advert_train <- tree(High_Sales_adver~.,Advertising_new,subset = train_advert)
plot(tree_advert_train)
text(tree_advert_train,pretty=0)
```



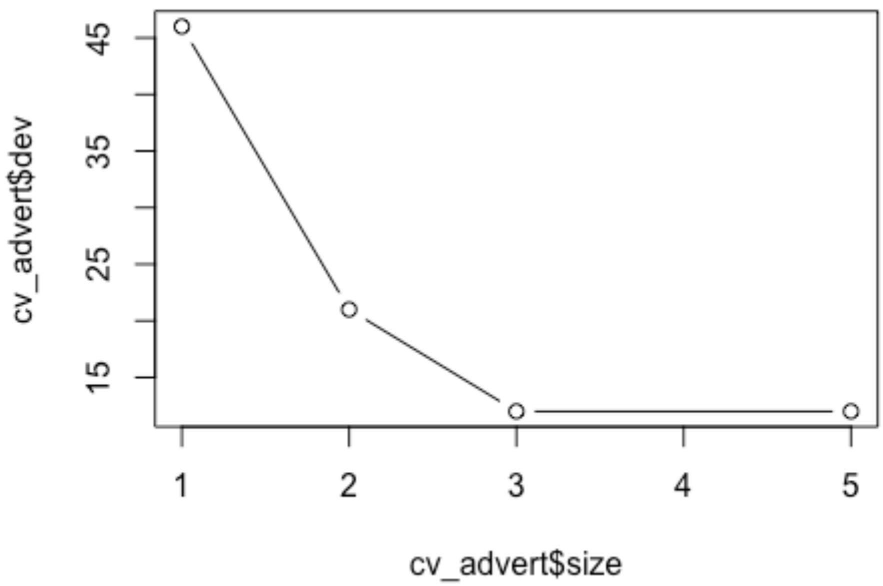
##

Test the model accuracy

```
tree_predicted_advert <- predict(tree_advert_train,test_advert,type="class")
High_test_advert <- High_Sales_adver[-train_advert]
table(tree_predicted_advert,High_test_advert)
##                  High_test_advert
## tree_predicted_advert No Yes
##                   No  53   2
##                   Yes  4  41
matrix_advert<-table(tree_predicted_advert,High_test_advert)
misrate<-((matrix_advert[1,2]+matrix_advert[2,1])/sum(matrix_advert))
paste("Misclassification error rate is ",misrate)
## [1] "Misclassification error rate is  0.06"
```

Construct the Cross validation plo

```
set.seed(3)
cv_advert <- cv.tree(tree_advert_train,FUN = prune.misclass)
names(cv_advert)
## [1] "size"   "dev"    "k"      "method"
cv_advert
## $size
## [1] 5 3 2 1
##
## $dev
## [1] 12 12 21 46
##
## $k
## [1] -Inf    0   11   30
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
plot(cv_advert$size,cv_advert$dev,type="b")
```



#

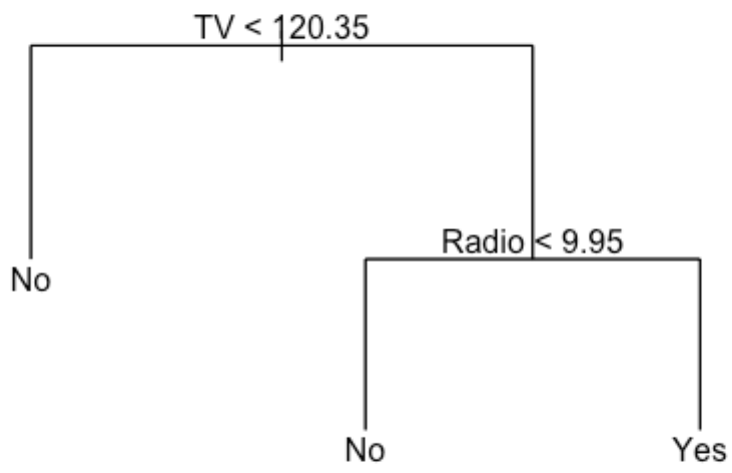
Select the best size of the tree model

```
print('the best size of the tree model is 3')
## [1] "the best size of the tree model is 3"
```

Obtain the best classification tree by pruning

```
pruned_advert <- prune.misclass(tree_advert_train,best=3)
pruned_advert
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
```

```
##
## 1) root 100 138.00 No ( 0.54000 0.46000 )
##   2) TV < 120.35 44  21.90 No ( 0.93182 0.06818 ) *
##   3) TV > 120.35 56  60.69 Yes ( 0.23214 0.76786 )
##     6) Radio < 9.95 11   0.00 No ( 1.00000 0.00000 ) *
##     7) Radio > 9.95 45  16.36 Yes ( 0.04444 0.95556 ) *
plot(pruned_advert)
text(pruned_advert,pretty=0)
```



#

Test the model accuracy

```
tree_predicted_advert_pruned <- predict(pruned_advert,test_advert,type="class")
High_test_advert_pruned <- High_Sales_adver[-train_advert]
table(tree_predicted_advert_pruned,High_test_advert_pruned)
##                             High_test_advert_pruned
## tree_predicted_advert_pruned No  Yes
##                          No  53   2
##                          Yes  4  41
matrix_advert_pruned <- table(tree_predicted_advert_pruned,High_test_advert_pruned)
mis_rate_pruned <-
((matrix_advert_pruned[1,2]+matrix_advert_pruned[2,1])/sum(matrix_advert_pruned))

paste("Misclassification error rate of Advertising is",mis_rate_pruned)
## [1] "Misclassification error rate of Advertising is 0.06"
```

Describe the terminal nodes of the resulting decision tree

if TV <120.35 then salse is low, but TV > 120.35 then the sales is higher