decision tree can be built with target variable Sale

library(readr)  
data<-read.csv("file:///E:/assignments data/descition tree/Company\_Data.csv")  
View(data)  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

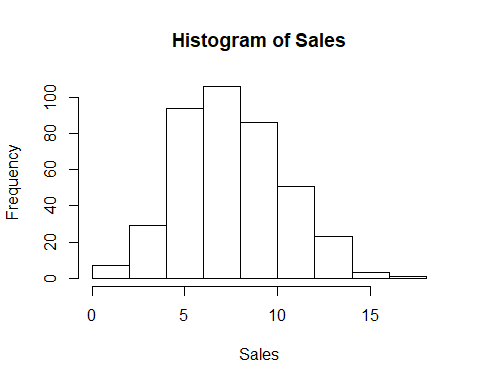
library(C50)  
str(data)

## 'data.frame': 400 obs. of 11 variables:  
## $ Sales : num 9.5 11.22 10.06 7.4 4.15 ...  
## $ CompPrice : int 138 111 113 117 141 124 115 136 132 132 ...  
## $ Income : int 73 48 35 100 64 113 105 81 110 113 ...  
## $ Advertising: int 11 16 10 4 3 13 0 15 0 0 ...  
## $ Population : int 276 260 269 466 340 501 45 425 108 131 ...  
## $ Price : int 120 83 80 97 128 72 108 120 124 124 ...  
## $ ShelveLoc : Factor w/ 3 levels "Bad","Good","Medium": 1 2 3 3 1 1 3 2 3 3 ...  
## $ Age : int 42 65 59 55 38 78 71 67 76 76 ...  
## $ Education : int 17 10 12 14 13 16 15 10 10 17 ...  
## $ Urban : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 1 ...  
## $ US : Factor w/ 2 levels "No","Yes": 2 2 2 2 1 2 1 2 1 2 ...

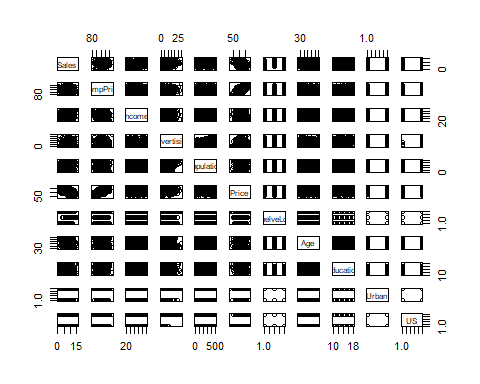
head(data)

## Sales CompPrice Income Advertising Population Price ShelveLoc Age Education  
## 1 9.50 138 73 11 276 120 Bad 42 17  
## 2 11.22 111 48 16 260 83 Good 65 10  
## 3 10.06 113 35 10 269 80 Medium 59 12  
## 4 7.40 117 100 4 466 97 Medium 55 14  
## 5 4.15 141 64 3 340 128 Bad 38 13  
## 6 10.81 124 113 13 501 72 Bad 78 16  
## Urban US  
## 1 Yes Yes  
## 2 Yes Yes  
## 3 Yes Yes  
## 4 Yes Yes  
## 5 Yes No  
## 6 No Yes

attach(data)  
hist( Sales)



pairs(data)



highsales1<-ifelse(Sales<10, "No", "Yes")  
data1<-data.frame(data, highsales1)  
inTraininglocal<-createDataPartition(highsales1, p=0.75, list=F )  
traindata<-data1[inTraininglocal, ]  
testdata<-data1[-inTraininglocal, ]  
names(traindata)

## [1] "Sales" "CompPrice" "Income" "Advertising" "Population"   
## [6] "Price" "ShelveLoc" "Age" "Education" "Urban"   
## [11] "US" "highsales1"

library(party)

## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Loading required package: zoo

##   
## Attaching package: 'zoo'

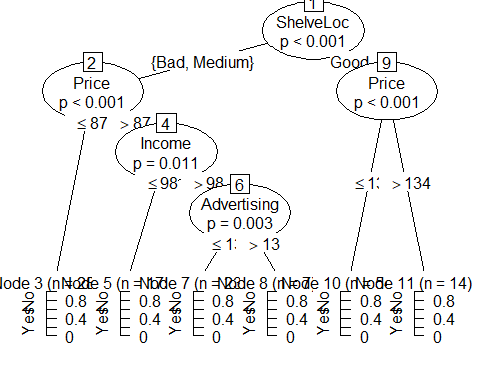
## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

## Loading required package: sandwich

optree=ctree(highsales1 ~ CompPrice+Income+Advertising+Population+Price+ShelveLoc+Age+Education+Urban+US, data=traindata)  
summary(optree)

## Length Class Mode   
## 1 BinaryTree S4

plot(optree)



pred<-predict(optree, testdata)  
head(pred)

## [1] No Yes Yes No Yes Yes  
## Levels: No Yes

a<-table(testdata$highsales1, pred)  
a

## pred  
## No Yes  
## No 70 10  
## Yes 9 10

confusionMatrix(testdata$highsales1, pred)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 70 10  
## Yes 9 10  
##   
## Accuracy : 0.8081   
## 95% CI : (0.7166, 0.8803)  
## No Information Rate : 0.798   
## P-Value [Acc > NIR] : 0.46   
##   
## Kappa : 0.3934   
##   
## Mcnemar's Test P-Value : 1.00   
##   
## Sensitivity : 0.8861   
## Specificity : 0.5000   
## Pos Pred Value : 0.8750   
## Neg Pred Value : 0.5263   
## Prevalence : 0.7980   
## Detection Rate : 0.7071   
## Detection Prevalence : 0.8081   
## Balanced Accuracy : 0.6930   
##   
## 'Positive' Class : No   
##

## accuracy=0.79 for better accuracies using boosting and bagging   
##boosting   
model<-C5.0(highsales1~CompPrice+Income+Advertising+Population+Price+ShelveLoc+Age+Education+Urban+US, data=traindata, tdrails=40)  
pred\_values<-predict.C5.0(model, testdata[-12])  
a<-table(testdata$highsales1, pred\_values)  
a

## pred\_values  
## No Yes  
## No 71 9  
## Yes 9 10

acc<-sum(diag(a)/sum(a))  
acc

## [1] 0.8181818

## bagging  
acc<-c()  
for(i in 1:100){  
 print(i)  
 intraininglocal1<-createDataPartition(data1$highsales1, p=0.80, list=F)  
 train<-data1[intraininglocal1,]  
 test<-data1[-intraininglocal1,]  
 model<-C5.0(train$highsales1~ CompPrice+Income+Advertising+Population+Price+ShelveLoc+Age+Education+Urban+US, data=train)  
 pred<-predict.C5.0(model, test[-12])  
 b<-table(test$highsales1, pred)  
 acc<-c(acc, sum(diag(b)/sum(b)))  
}

## [1] 1  
## [1] 2  
## [1] 3  
## [1] 4  
## [1] 5  
## [1] 6  
## [1] 7  
## [1] 8  
## [1] 9  
## [1] 10  
## [1] 11  
## [1] 12  
## [1] 13  
## [1] 14  
## [1] 15  
## [1] 16  
## [1] 17  
## [1] 18  
## [1] 19  
## [1] 20  
## [1] 21  
## [1] 22  
## [1] 23  
## [1] 24  
## [1] 25  
## [1] 26  
## [1] 27  
## [1] 28  
## [1] 29  
## [1] 30  
## [1] 31  
## [1] 32  
## [1] 33  
## [1] 34  
## [1] 35  
## [1] 36  
## [1] 37  
## [1] 38  
## [1] 39  
## [1] 40  
## [1] 41  
## [1] 42  
## [1] 43  
## [1] 44  
## [1] 45  
## [1] 46  
## [1] 47  
## [1] 48  
## [1] 49  
## [1] 50  
## [1] 51  
## [1] 52  
## [1] 53  
## [1] 54  
## [1] 55  
## [1] 56  
## [1] 57  
## [1] 58  
## [1] 59  
## [1] 60  
## [1] 61  
## [1] 62  
## [1] 63  
## [1] 64  
## [1] 65  
## [1] 66  
## [1] 67  
## [1] 68  
## [1] 69  
## [1] 70  
## [1] 71  
## [1] 72  
## [1] 73  
## [1] 74  
## [1] 75  
## [1] 76  
## [1] 77  
## [1] 78  
## [1] 79  
## [1] 80  
## [1] 81  
## [1] 82  
## [1] 83  
## [1] 84  
## [1] 85  
## [1] 86  
## [1] 87  
## [1] 88  
## [1] 89  
## [1] 90  
## [1] 91  
## [1] 92  
## [1] 93  
## [1] 94  
## [1] 95  
## [1] 96  
## [1] 97  
## [1] 98  
## [1] 99  
## [1] 100

summary(acc)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.7468 0.8228 0.8481 0.8391 0.8608 0.9241

acc

## [1] 0.7974684 0.8734177 0.8227848 0.7848101 0.8481013 0.8227848 0.8734177  
## [8] 0.8481013 0.8101266 0.7721519 0.8354430 0.8860759 0.7721519 0.8481013  
## [15] 0.8607595 0.8481013 0.8354430 0.8354430 0.8607595 0.8734177 0.8860759  
## [22] 0.7848101 0.8481013 0.8354430 0.8481013 0.8607595 0.8101266 0.8481013  
## [29] 0.8354430 0.8734177 0.8481013 0.8860759 0.8227848 0.8607595 0.8734177  
## [36] 0.8607595 0.8481013 0.8227848 0.8860759 0.8734177 0.8354430 0.8227848  
## [43] 0.8607595 0.8101266 0.8481013 0.7721519 0.8101266 0.8860759 0.8481013  
## [50] 0.8227848 0.7468354 0.8227848 0.8227848 0.9240506 0.7974684 0.8860759  
## [57] 0.8607595 0.7974684 0.8227848 0.8354430 0.8354430 0.8987342 0.8354430  
## [64] 0.8607595 0.8101266 0.8607595 0.8734177 0.8354430 0.8481013 0.8734177  
## [71] 0.8227848 0.8481013 0.8607595 0.8101266 0.7721519 0.8227848 0.7848101  
## [78] 0.8101266 0.7974684 0.8860759 0.7594937 0.8734177 0.8227848 0.8481013  
## [85] 0.7721519 0.8354430 0.8481013 0.8481013 0.8481013 0.9113924 0.8607595  
## [92] 0.8860759 0.8607595 0.8101266 0.8101266 0.8481013 0.8607595 0.8607595  
## [99] 0.8354430 0.8101266