Decision Trees in R, Decision trees are mainly classification and regression types.

Classification means Y variable is factor and regression type means Y variable is numeric.

Just look at one of the examples from each type,

Classification example is detecting email spam data and regression tree example is from Boston housing data.

Decision trees are also called Trees and CART.

CART indicates classification and regression trees.

The main goal behind classification tree is to classify or predict an outcome based on a set of predictors.

**Advantageous of Decision Trees**

Easy Interpretation

Making prediction is fast

Easy to identify important variables

Handless missing data

One of the drawbacks is to can have high variability in performance.

Recursive portioning- basis can achieve maximum homogeneity within the new partition.

**Decision Trees in R**

**Method 1:- Classification Tree**

**Load Library**

library(DAAG)

library(party)

library(rpart)

library(rpart.plot)

library(mlbench)

library(caret)

library(pROC)

library(tree)

**Getting Data -Email Spam Detection**

str(spam7)

data.frame':  4601 obs. of  7 variables:

 $ crl.tot: num  278 1028 2259 191 191 ...

 $ dollar : num  0 0.18 0.184 0 0 0 0.054 0 0.203 0.081 ...

 $ bang   : num  0.778 0.372 0.276 0.137 0.135 0 0.164 0 0.181 0.244 ...

 $ money  : num  0 0.43 0.06 0 0 0 0 0 0.15 0 ...

 $ n000   : num  0 0.43 1.16 0 0 0 0 0 0 0.19 ...

 $ make   : num  0 0.21 0.06 0 0 0 0 0 0.15 0.06 ...

 $ yesno  : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...

Total 4601 observations and 7 variables.

mydata <- spam7

**Data Partition**

set.seed(1234)

ind <- sample(2, nrow(mydata), replace = T, prob = c(0.5, 0.5))

train <- mydata[ind == 1,]

test <- mydata[ind == 2,]

Tree Classification

tree <- rpart(yesno ~., data = train)

rpart.plot(tree)

printcp(tree)

Classification tree:

rpart(formula = yesno ~ ., data = train)

Variables actually used in tree construction:

[1] bang    crl.tot dollar

Root node error: 900/2305 = 0.39046

n= 2305

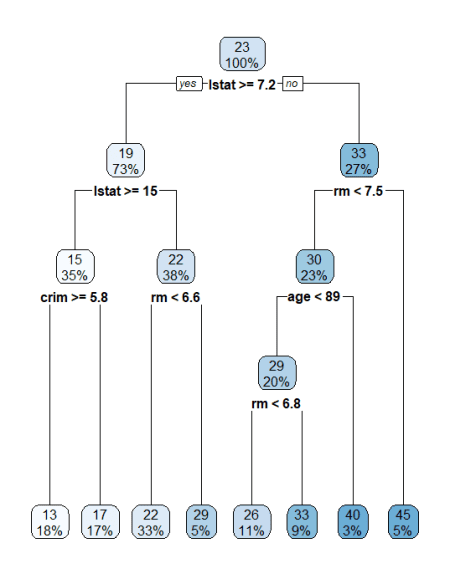
        CP nsplit rel error  xerror     xstd

1 0.474444      0   1.00000 1.00000 0.026024

2 0.074444      1   0.52556 0.56556 0.022128

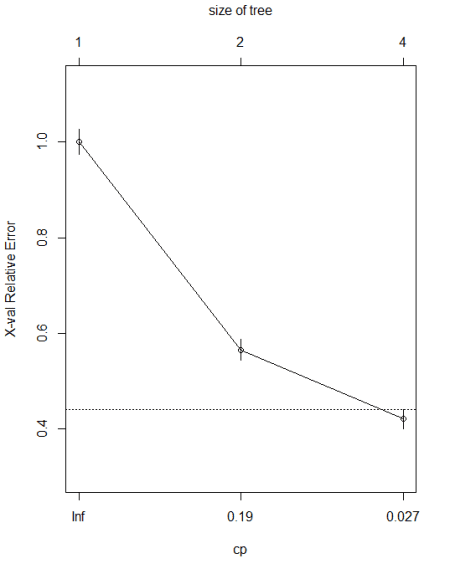
3 0.010000      3   0.37667 0.42111 0.019773

plotcp(tree)



You can change the cp value according to your data set. Please note lower cp value means bigger the tree. If you are using too lower cp that leads to overfitting also.

tree <- rpart(yesno ~., data = train,cp=0.07444)



**Confusion matrix -train**

p <- predict(tree, train, type = 'class')

confusionMatrix(p, train$yesno, positive=’y’)

Please make sure you mention positive classes in the confusion matrix.

Confusion Matrix and Statistics

          Reference

Prediction    n    y

         n 1278  212

         y  127  688

               Accuracy : 0.8529

                 95% CI : (0.8378, 0.8671)

    No Information Rate : 0.6095

    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6857

 Mcnemar's Test P-Value : 5.061e-06

           Sensitivity : 0.7644

            Specificity : 0.9096

         Pos Pred Value : 0.8442

         Neg Pred Value : 0.8577

             Prevalence : 0.3905

         Detection Rate : 0.2985

   Detection Prevalence : 0.3536

      Balanced Accuracy : 0.8370

       'Positive' Class : y

Model has 85% accuracy

**ROC**

p1 <- predict(tree, test, type = 'prob')

p1 <- p1[,2]

r <- multiclass.roc(test$yesno, p1, percent = TRUE)

roc <- r[['rocs']]

r1 <- roc[[1]]

plot.roc(r1,

         print.auc=TRUE,

         auc.polygon=TRUE,

         grid=c(0.1, 0.2),

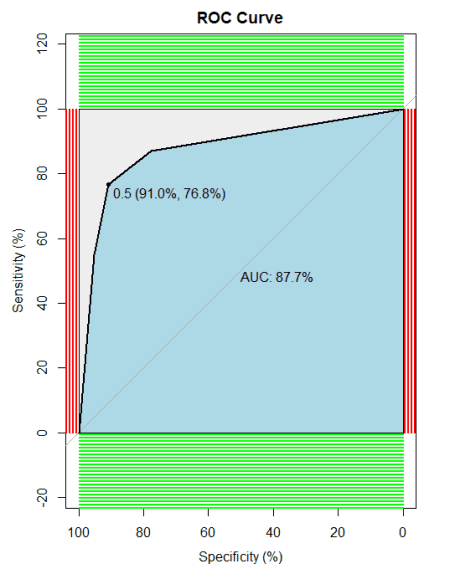
         grid.col=c("green", "red"),

         max.auc.polygon=TRUE,

         auc.polygon.col="lightblue",

         print.thres=TRUE,

         main= 'ROC Curve')



**Method 2- Regression  Tree**

data('BostonHousing')

mydata <- BostonHousing

**Data Partition**

set.seed(1234)

ind <- sample(2, nrow(mydata), replace = T, prob = c(0.5, 0.5))

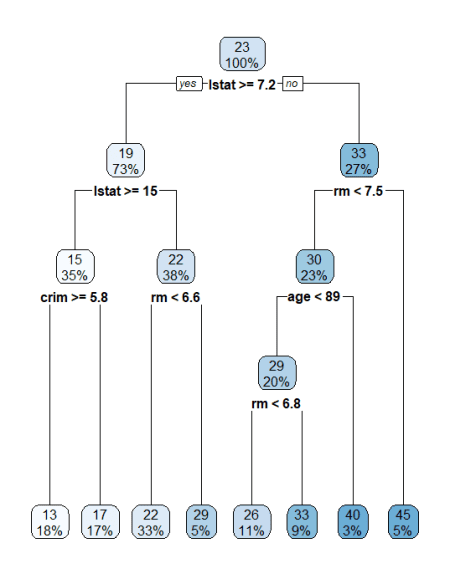
train <- mydata[ind == 1,]

test <- mydata[ind == 2,]

Regression tree

tree <- rpart(medv ~., data = train)

rpart.plot(tree)



printcp(tree)

Regression tree:

rpart(formula = medv ~ ., data = train)

Variables actually used in tree construction:

[1] age   crim  lstat rm

Root node error: 22620/262 = 86.334

n= 262

        CP nsplit rel error  xerror     xstd

0.469231      0   1.00000 1.01139 0.115186

2 0.128700      1   0.53077 0.62346 0.080154

3 0.098630      2   0.40207 0.51042 0.076055

4 0.033799      3   0.30344 0.42674 0.069827

5 0.028885      4   0.26964 0.39232 0.066342

6 0.028018      5   0.24075 0.37848 0.066389

7 0.015141      6   0.21274 0.34877 0.065824

8 0.010000      7   0.19760 0.33707 0.065641

rpart.rules(tree)

medv

13 when lstat >=        14.8 & crim >= 5.8

17 when lstat >=        14.8 & crim <  5.8

22 when lstat is 7.2 to 14.8 & rm <  6.6

26 when lstat <  7.2         & rm <  6.8        & age <  89

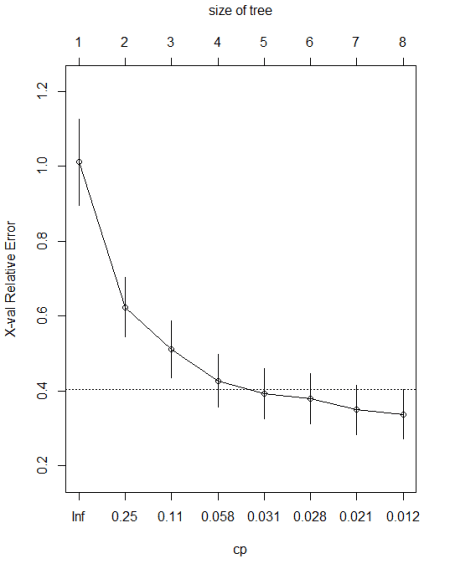
29 when lstat is 7.2 to 14.8 & rm >=        6.6

33 when lstat <  7.2         & rm is 6.8 to 7.5 & age <  89

40 when lstat <  7.2         & rm <  7.5        & age >= 89

45 when lstat <  7.2         & rm >=        7.5

plotcp(tree)



**Predict**

p <- predict(tree, train)

**Root Mean Square Error**

sqrt(mean((train$medv-p)^2))

4.130294

**R Square**

(cor(train$medv,p))^2

0.8024039

**Conclusion**

In the regression model, the r square value is 80% and RMSE is 4.13, not bad at all..In this way, you can make use of Decision classification regression tree models.