R-Lab: Decision Trees

require(mlbench)  
require(rpart)  
library(help = "datasets")  
  
data(library(help = "datasets"))  
  
# example of entropy  
p <- 4/6  
e0 <- -p \* log2(p) - (1 - p) \* log2(1 - p)  
e1 <- -2/3 \* log2(2/3) - 1/3 \* log2(1/3)  
e0 - 0.5 \* e1

## [1] 0.4591479

# pruning  
estimate <- function(e, n) {  
 return(e + 1.15 \* sqrt(e \* (1 - e)/n))  
}  
estimate(1, 2)

## [1] 1

a <- estimate(9/19, 19) \* 19  
b <- estimate(9/20, 20) \* 20  
estimate(19/39, 39) \* 39

## [1] 22.58969

a + b

## [1] 23.06148

estimate(2/12, 12) \* 12

## [1] 3.484644

estimate(21/51, 51) \* 51

## [1] 25.04188

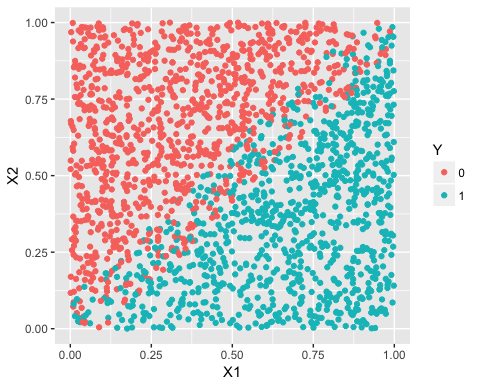
estimate(0/10, 10) \* 10

## [1] 0

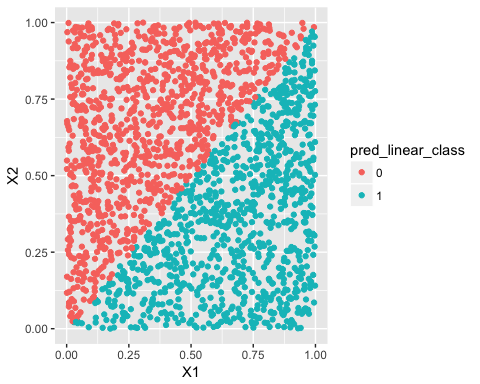
estimate(19/49, 49) \* 49

## [1] 22.92227

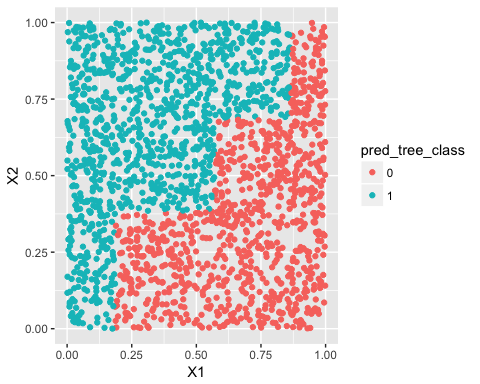
# comparison between Decision tree and regression model  
require(rpart)  
require(dplyr)  
require(ggplot2)  
ndata <- 2000  
X1 <- runif(ndata, min = 0, max = 1)  
X2 <- runif(ndata, min = 0, max = 1)  
data <- data.frame(X1, X2)  
data <- data %>% mutate(X12 = 0.5 \* (X1 - X2), Y = ifelse(X12 >= 0, 1, 0))  
ix <- which(abs(data$X12) <= 0.05)  
data$Y[ix] <- ifelse(runif(length(ix)) < 0.5, 0, 1)  
data <- data %>% select(-X12) %>% mutate(Y = as.factor(as.character(Y)))  
ggplot(data, aes(x = X1, y = X2, color = Y)) + geom\_point()



linear\_model <- glm(Y ~ ., family = binomial(link = "logit"), data = data)  
tree\_model <- rpart(Y ~ ., data = data)  
pred\_linear <- predict(linear\_model, data, type = "response")  
pred\_tree <- predict(tree\_model, data, type = "prob")[, 1]  
data\_pred <- data %>% mutate(pred\_linear\_class = ifelse(pred\_linear < 0.5, 0,   
 1)) %>% mutate(pred\_linear\_class = as.factor(as.character(pred\_linear\_class))) %>%   
 mutate(pred\_tree\_class = ifelse(pred\_tree < 0.5, 0, 1)) %>% mutate(pred\_tree\_class = as.factor(as.character(pred\_tree\_class)))  
ggplot(data\_pred, aes(x = X1, y = X2, color = pred\_linear\_class)) + geom\_point()



ggplot(data\_pred, aes(x = X1, y = X2, color = pred\_tree\_class)) + geom\_point()



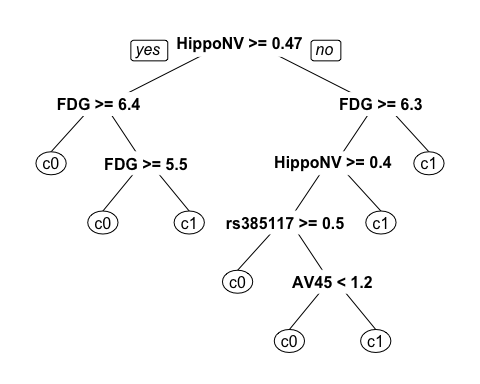
## R-lab

Here we use DX\_bl as the target variable, and use variables other than ID, TOTAL13 and MMSCORE to predict DX\_bl. Since DX\_bl is binary, so this can considered as a classification problem.

library(rpart)  
library(rpart.plot)  
library(dplyr)  
library(tidyr)  
library(ggplot2)  
library(RCurl)  
library(partykit)  
theme\_set(theme\_gray(base\_size = 15))  
  
  
data <- read.csv(text = getURL("https://raw.githubusercontent.com/shuailab/ind\_498/master/resource/data/AD.csv"))  
  
target\_indx <- which(colnames(data) == "DX\_bl")  
data[, target\_indx] <- as.factor(paste0("c", data[, target\_indx]))  
rm\_indx <- which(colnames(data) %in% c("ID", "TOTAL13", "MMSCORE"))  
data <- data[, -rm\_indx]

Apply a decision tree to the data and plot the decision tree.

tree <- rpart(DX\_bl ~ ., data)  
# rpart.plot(tree,fallen.leaves=FALSE,type=0,xcompact=TRUE,compact=TRUE)  
prp(tree, nn.cex = 1)

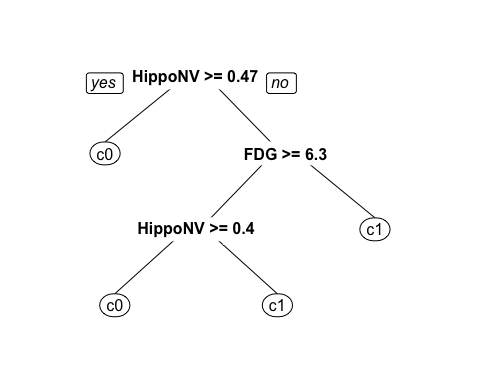
 The importance score for each variable can be obtained from the tree. HippoNV has the largest importance score among all variables.

print(tree$variable.importance)

## HippoNV FDG AV45 AGE PTGENDER e4\_1   
## 116.6665538 89.5608444 39.9595988 28.2195180 12.2040648 6.4708596   
## rs3851179 PTEDUCAT rs3818361   
## 4.2352941 1.1552265 0.8663915

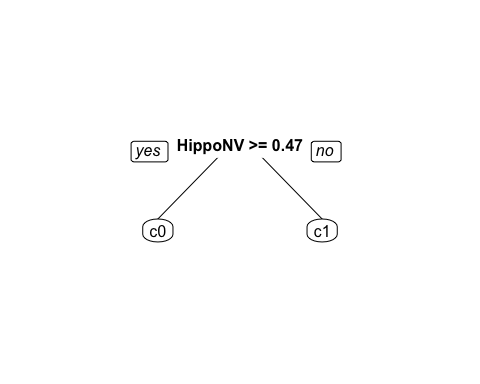
The tree can further pruned with the prune function and parameter cp to control the complexity. cp is the minimum relative error improved by splitting the node. A larger cp leads to a less-complex tree. First we try 0.01, the number of nodes is reduced.

tree\_0.05 <- prune(tree, cp = 0.05)  
# rpart.plot(tree\_0.05,fallen.leaves=FALSE)  
prp(tree\_0.05, nn.cex = 1)



Then we increase cp to 0.1. The tree only has two nodes. Here we demonstrate the pruning process by using parameter cp. In practicse, cp can be decided by minimizing the cross-validation error.

tree\_0.1 <- prune(tree, cp = 0.1)  
# rpart.plot(tree\_0.1,fallen.leaves=FALSE)  
prp(tree\_0.1, nn.cex = 1)

 We adjust the minimum node size of decision tree from 200 to 1, so the decision trees' complexity is expected to increase. Half of the data points for training a decision tree, and the other half is used for testing. A training error and testing error can be calculated for each tree structure. For each tree, the number of leaf nodes is recorded and used for measuring the complexity of the tree.

set.seed(1)  
train.ix <- sample(nrow(data), floor(nrow(data)/2))  
err.train.v <- NULL  
err.test.v <- NULL  
leaf.v <- NULL  
for (i in seq(0.2, 0, by = -0.005)) {  
 tree <- rpart(DX\_bl ~ ., data = data[train.ix, ], cp = i)  
 pred.train <- predict(tree, data[train.ix, ], type = "class")  
 pred.test <- predict(tree, data[-train.ix, ], type = "class")  
 current.err.train <- length(which(pred.train != data[train.ix, ]$DX\_bl))/length(pred.train)  
 current.err.test <- length(which(pred.test != data[-train.ix, ]$DX\_bl))/length(pred.test)  
 err.train.v <- c(err.train.v, current.err.train)  
 err.test.v <- c(err.test.v, current.err.test)  
 leaf.v <- c(leaf.v, length(which(tree$frame$var == "<leaf>")))  
}  
err.mat <- as.data.frame(cbind(train\_err = err.train.v, test\_err = err.test.v,   
 leaf\_num = leaf.v))  
err.mat$leaf\_num <- as.factor(err.mat$leaf\_num)  
err.mat <- unique(err.mat)  
err.mat <- err.mat %>% gather(type, error, train\_err, test\_err)

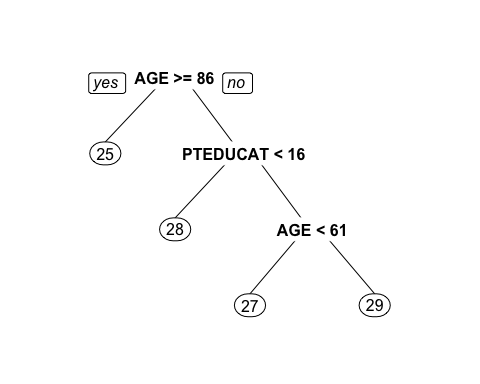
The training errors and testing errors of the trees at different number of leaf nodes are plotted. It can be seen that, as the complexity of trees increases, the training errors continue to decrease, while the testing errors first decrease but increase at some point. This indicates that testing error should be used for measuring a decision tree as minimizing the training error can lead to overfitting.

data.plot <- err.mat %>% mutate(type = factor(type))  
ggplot(data.plot, aes(x = leaf\_num, y = error, shape = type, color = type)) +   
 geom\_line() + geom\_point(size = 3)



We have been focused on classification problems so far, now let's also try rpart for the same regression problem illusrated in the linear regression R lab. Firstly, AGE, PTGENDER and PTEDUCAT are used as the predictor variables. The tree is plotted in the Figure below. The prediction of MMSCORE (a numeric value) is labeled at each leaf node. In the linear model part, it has been shown that the relationship between MMSCORE and PTEDUCAT changes substantially according to different levels of AGE. The decision tree is able to capture the interaction between PTEDUCAT, AGE and MMSCORE.

AD <- read.csv(text = getURL("https://raw.githubusercontent.com/shuailab/ind\_498/master/resource/data/AD2.csv"))  
AD\_demo <- subset(AD, select = c("MMSCORE", "AGE", "PTGENDER", "PTEDUCAT"))  
tree <- rpart(MMSCORE ~ ., AD\_demo, method = "anova")  
# rpart.plot(tree,fallen.leaves=FALSE,type=0,xcompact=TRUE)  
prp(tree, nn.cex = 1)



# print(tree) printcp(tree)

Now let's build a decision tree with all predictor variables. It can be seen that more interactions are captured. The tree can also provide insight for feature engineering in a linear modeling context, e.g., now we can add these interactions, such as FDG and HippoNV, HippoNV and AGE, as new features to the linear regression model and evaluate the incremental accuracy gain. However, it should noted that the interactions useful in a decision tree may not be optimal for a linear model given the models are built with different evaluation criteria.

AD\_full <- AD[, c(1:16)]  
tree <- rpart(MMSCORE ~ ., AD\_full, method = "anova")  
prp(tree, nn.cex = 1)

