Decision Trees

# Decision tree is a type of supervised learning algorithm with a pre-defined target variable used for classifcation problems.

## There are two types:

* Categorical Variable Decision tree + Student will play baseball: Y or N
* Continous Variable Decision Tree

## Advantages

* Easy
* Used in data exploration
* Less data cleaning required
* Data type is not a contstraint
* Non parametric method

## Disadvantages

* Overfitting
* Not fit for continous variables

## Node splits based on algorithms

1. Gini Index

* Works well with categorical target: 1 or 0 +Binary Split
* CART uses Gini method to create binary splits
* For sub-nodes, uses sum of squares: P^2 + Q^2

1. Chi-Square

* Uses Chi Square
* Generates tree call CHAID
* To find out the statistical significance between the differences between sub-nodes and parent node

1. Information Gain
2. Reduction in Variance

## Key Parameters

1. Min\_samples\_split

* Defines the minimum number of samples (or observations) which are required in a node to be considered for splitting.
* Used to control over-fitting. Higher values prevent a model from learning relations which might be highly specific to the particular sample selected for a tree.
* Too high values can lead to under-fitting hence, it should be tuned using CV.

1. Min\_sample\_leaf

* Defines the minimum samples (or observations) required in a terminal node or leaf.
* Used to control over-fitting similar to min\_samples\_split.
* Generally lower values should be chosen for imbalanced class problems because the regions in which the minority class will be in majority will be very small
* The maximum depth of a tree.
* Used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample.
* Should be tuned using CV.

1. Max\_features

* The maximum number of terminal nodes or leaves in a tree.
* Can be defined in place of max\_depth. Since binary trees are created, a depth of ‘n’ would produce a maximum of 2^n leaves.

## Need to prune tree

* We first make the decision tree to a large depth.
* Then we start at the bottom and start removing leaves which are giving us negative returns when compared from the top.
* Suppose a split is giving us a gain of say -10 (loss of 10) and then the next split on that gives us a gain of 20. A simple decision tree will stop at step 1 but in pruning, we will see that the overall gain is +10 and keep both leaves.

# Quick Examples of Decision Trees

Example 1: Titanic Data Set From Kaggle

#Data Set Up  
train <- read.csv("train.csv", stringsAsFactors=FALSE)  
test <- read.csv("test.csv", stringsAsFactors=FALSE)  
attach(test)  
attach(train)

## The following objects are masked from test:  
##   
## Age, Cabin, Embarked, Fare, Name, Parch, PassengerId, Pclass,  
## Sex, SibSp, Ticket

#Look at data  
head(train)

## PassengerId Survived Pclass  
## 1 1 0 3  
## 2 2 1 1  
## 3 3 1 3  
## 4 4 1 1  
## 5 5 0 3  
## 6 6 0 3  
## Name Sex Age SibSp  
## 1 Braund, Mr. Owen Harris male 22 1  
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female 38 1  
## 3 Heikkinen, Miss. Laina female 26 0  
## 4 Futrelle, Mrs. Jacques Heath (Lily May Peel) female 35 1  
## 5 Allen, Mr. William Henry male 35 0  
## 6 Moran, Mr. James male NA 0  
## Parch Ticket Fare Cabin Embarked  
## 1 0 A/5 21171 7.2500 S  
## 2 0 PC 17599 71.2833 C85 C  
## 3 0 STON/O2. 3101282 7.9250 S  
## 4 0 113803 53.1000 C123 S  
## 5 0 373450 8.0500 S  
## 6 0 330877 8.4583 Q

str(train)

## 'data.frame': 891 obs. of 12 variables:  
## $ PassengerId: int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Survived : int 0 1 1 1 0 0 0 0 1 1 ...  
## $ Pclass : int 3 1 3 1 3 3 1 3 3 2 ...  
## $ Name : chr "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques Heath (Lily May Peel)" ...  
## $ Sex : chr "male" "female" "female" "female" ...  
## $ Age : num 22 38 26 35 35 NA 54 2 27 14 ...  
## $ SibSp : int 1 1 0 1 0 0 0 3 0 1 ...  
## $ Parch : int 0 0 0 0 0 0 0 1 2 0 ...  
## $ Ticket : chr "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...  
## $ Fare : num 7.25 71.28 7.92 53.1 8.05 ...  
## $ Cabin : chr "" "C85" "" "C123" ...  
## $ Embarked : chr "S" "C" "S" "S" ...

#Percent of those who survived  
table(train$Survived)

##   
## 0 1   
## 549 342

prop.table(table(train$Survived)) ###38% Survived

##   
## 0 1   
## 0.6161616 0.3838384

#Everyone dies prediction  
test$Survived <- rep(0,418)  
### rep function will create the survived column in data frame and repeat  
###'0' prediction 418  
### very simple prediction  
  
#Create new data frame with Pass ID and survival  
submit <- data.frame(PassengerId = test$PassengerId, Survived = test$Survived)  
write.csv(submit, file = "theyallperish.csv", row.names = FALSE)  
  
  
#Sex and Survival  
summary(train$Sex)

## Length Class Mode   
## 891 character character

table(train$Sex)

##   
## female male   
## 314 577

prop.table(table(train$Sex,train$Survived)) #divides by total num of passenger not within groups

##   
## 0 1  
## female 0.09090909 0.26150393  
## male 0.52525253 0.12233446

prop.table(table(train$Sex,train$Survived),1) #denominator with gender ##most women survived

##   
## 0 1  
## female 0.2579618 0.7420382  
## male 0.8110919 0.1889081

##New variable for model  
test$Survived <- 0  
test$Survived[test$Sex == 'female'] <- 1  
  
#Age and survival  
summary(train$Age)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 0.42 20.12 28.00 29.70 38.00 80.00 177

##Categorical assigments to Age  
train$Child <- 0  
train$Child[train$Age < 18] <- 1  
  
#Find the number of survivors for the Age and Gender Subsets  
aggregate(Survived ~ Child + Sex, data=train, FUN=sum)

## Child Sex Survived  
## 1 0 female 195  
## 2 1 female 38  
## 3 0 male 86  
## 4 1 male 23

##For percent  
aggregate(Survived ~ Child + Sex, data=train, FUN=function(x) {sum(x)/length(x)})

## Child Sex Survived  
## 1 0 female 0.7528958  
## 2 1 female 0.6909091  
## 3 0 male 0.1657033  
## 4 1 male 0.3965517

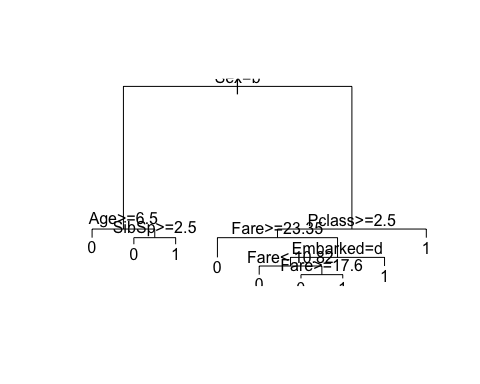
#Fare paid + Survival  
train$Fare2 <- '30+'  
train$Fare2[train$Fare < 30 & train$Fare >= 20] <- '20-30'  
train$Fare2[train$Fare < 20 & train$Fare >= 10] <- '10-20'  
train$Fare2[train$Fare < 10] <- '<10'  
  
#Class/Fare/Sex affects   
aggregate(Survived ~ Fare2 + Pclass + Sex, data=train, FUN=function(x) {sum(x)/length(x)})

## Fare2 Pclass Sex Survived  
## 1 20-30 1 female 0.8333333  
## 2 30+ 1 female 0.9772727  
## 3 10-20 2 female 0.9142857  
## 4 20-30 2 female 0.9000000  
## 5 30+ 2 female 1.0000000  
## 6 <10 3 female 0.5937500  
## 7 10-20 3 female 0.5813953  
## 8 20-30 3 female 0.3333333  
## 9 30+ 3 female 0.1250000  
## 10 <10 1 male 0.0000000  
## 11 20-30 1 male 0.4000000  
## 12 30+ 1 male 0.3837209  
## 13 <10 2 male 0.0000000  
## 14 10-20 2 male 0.1587302  
## 15 20-30 2 male 0.1600000  
## 16 30+ 2 male 0.2142857  
## 17 <10 3 male 0.1115385  
## 18 10-20 3 male 0.2368421  
## 19 20-30 3 male 0.1250000  
## 20 30+ 3 male 0.2400000

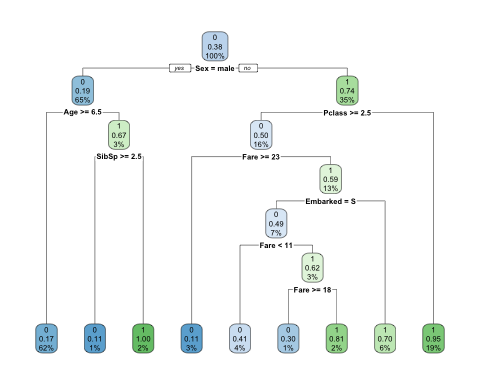
## Why would someone in third class with an expensive ticket be worse off?  
test$Survived <- 0  
test$Survived[test$Sex == 'female'] <- 1  
test$Survived[test$Sex == 'female' & test$Pclass == 3 & test$Fare >= 20] <- 0  
   
#Recursive Partitioning and Regression Trees  
library(rpart)

## Warning: package 'rpart' was built under R version 3.4.3

## to predict a continous variable use method = "anova"  
fit <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked,  
data=train,method="class")  
plot(fit)  
text(fit)



#For prettier tree  
library(rpart.plot)  
library(RColorBrewer)  
rpart.plot(fit)



# Random Forest gives more accurate predictions.