ASML Project - Exercise 2 - Titanic analysis - José Lise - DSTI S19

The goal is to carry out the Titanic data classification analysis. We will use the Titanic dataset available on Kaggle web site. In detail, this is a binary classification problem. The model must be able to predict survival or not with a good accuracy on the test sample.

The data has been splitted into two groups:

- training set (train.csv)
- test set (test.csv)

The training set will be used to build the machine learning models. For the training set, is provided the outcome (also known as the "ground truth") for each passenger.

The test set will be used to see how well the models perform on unseen data. For the test set, the ground truth for each passenger is not provided. It is the models' job to predict these outcomes. For each passenger in the test set, we will use the trained model to predict whether or not they survived the sinking of the Titanic. And as we don't have the outcome for the test set, we will submit our prediction to the kaggle web site to get our score.

Loading the data

```
setwd("D:/OneDrive - Data ScienceTech Institute/DSTI/AdvanceStatisticsMachineLearning/Project")
train <- read.csv("titanic/train.csv", stringsAsFactors=FALSE, header=TRUE, sep=',')
test <-  read.csv("titanic/test.csv", stringsAsFactors=FALSE, header=TRUE, sep=',')
```

check the train data frame

```
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)"
##  $ 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" ...
```

There are 891 observations of 12 variables. 5 variables are integers, 5 are characters and 2 are numeric.

Summary train

```
summary(train)
```

```
##   PassengerId       Survived          Pclass          Name
##  Min.   :  1.0   Min.   :0.0000   Min.   :1.000   Length:891
```

```
##  1st Qu.:223.5   1st Qu.:0.0000   1st Qu.:2.000   Class :character
##  Median :446.0   Median :0.0000   Median :3.000   Mode  :character
##  Mean   :446.0   Mean   :0.3838   Mean   :2.309
##  3rd Qu.:668.5   3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :891.0   Max.   :1.0000   Max.   :3.000
##
##      Sex                  Age            SibSp           Parch
##  Length:891         Min.   : 0.42   Min.   :0.000   Min.   :0.0000
##  Class :character   1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000
##  Mode  :character   Median :28.00   Median :0.000   Median :0.0000
##                     Mean   :29.70   Mean   :0.523   Mean   :0.3816
##                     3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##                     Max.   :80.00   Max.   :8.000   Max.   :6.0000
##                     NA's   :177
##     Ticket              Fare            Cabin             Embarked
##  Length:891         Min.   :  0.00   Length:891         Length:891
##  Class :character   1st Qu.:  7.91   Class :character   Class :character
##  Mode  :character   Median : 14.45   Mode  :character   Mode  :character
##                     Mean   : 32.20
##                     3rd Qu.: 31.00
##                     Max.   :512.33
##
```

The summary above already shows that there are 177 missing rows for the age variable.

Variables description

Here are the short description of the variables in the dataset:

- PassengerId: Identification number for passengers

- Survived: Indicates if the passenger survived: 0=NO, 1=YES

- Pclass: Ticket Class: 1=1st, 2=2nd, 3=3rd

- Sex: Female, Male

- Age: Age in years
- SibSp: # of sibling/Spouses abroard the Titanic
- Parch: # of Parents/Children abroad the Titanic
- Ticket: Ticket number
- fare: Passenger fare
- cabin: Cabin number
- embarked: Port of Embarkation: C=Cherburg, Q=Queenstown, S=Southampton

Here are some additionnal information for the variables: *pclass*: A proxy for socio-economic status (SES)

- 1st = Upper
- 2nd = Middle

- 3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way:

- Sibling = brother, sister, stepbrother, stepsister

- Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way:

- Parent = mother, father
- Child = daughter, son, stepdaughter, stepson
  Some children travelled only with a nanny, therefore parch=0 for them.

check the test data frame

```r
str(test)
```

```
## 'data.frame':    418 obs. of  11 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : chr  "Kelly, Mr. James" "Wilkes, Mrs. James (Ellen Needs)" "Myles, Mr. Thomas Franci
##  $ Sex        : chr  "male" "female" "male" "male" ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
##  $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ Ticket     : chr  "330911" "363272" "240276" "315154" ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : chr  "" "" "" "" ...
##  $ Embarked   : chr  "Q" "S" "Q" "S" ...
```

Test data set contains 418 observation of 11 variables. As expected, the survived variable is missing from this data set.

Test summary

```r
summary(test)
```

```
##   PassengerId         Pclass          Name               Sex           
##  Min.   : 892.0   Min.   :1.000   Length:418         Length:418        
##  1st Qu.: 996.2   1st Qu.:1.000   Class :character   Class :character  
##  Median :1100.5   Median :3.000   Mode  :character   Mode  :character  
##  Mean   :1100.5   Mean   :2.266                                        
##  3rd Qu.:1204.8   3rd Qu.:3.000                                        
##  Max.   :1309.0   Max.   :3.000                                        
##                                                                        
##       Age            SibSp            Parch           Ticket         
##  Min.   : 0.17   Min.   :0.0000   Min.   :0.0000   Length:418        
##  1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.0000   Class :character  
##  Median :27.00   Median :0.0000   Median :0.0000   Mode  :character  
##  Mean   :30.27   Mean   :0.4474   Mean   :0.3923                     
##  3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.0000                     
##  Max.   :76.00   Max.   :8.0000   Max.   :9.0000                     
##  NA's   :86                                                          
##       Fare             Cabin             Embarked        
##  Min.   :  0.000   Length:418         Length:418        
##  1st Qu.:  7.896   Class :character   Class :character  
##  Median : 14.454   Mode  :character   Mode  :character  
##  Mean   : 35.627                                        
##  3rd Qu.: 31.500                                        
##  Max.   :512.329                                        
##  NA's   :1
```

keep raw train, and test data sets for future use during the modeling part. However we will transform the variables Pclass, Sex and Embarked to factors.

```
train_raw <- train
test_raw <- test

train_raw$Pclass <- factor(train_raw$Pclass)
train_raw$Sex <- factor(train_raw$Sex)
train_raw$Embarked <- factor(train_raw$Embarked, exclude="")

test_raw$Pclass <- factor(test_raw$Pclass)
test_raw$Sex <- factor(test_raw$Sex)
test_raw$Embarked <- factor(test_raw$Embarked, exclude="")

test_raw$Survived <- 0
all_raw <- rbind(train_raw,test_raw)
```

Merge train and test data set for exploratory analysis

```
# Create a Survided column for the test dataset anf fill it with 0
test$Survived <- 0
all <- rbind(train,test)
```

Handling Missing Data

```
summary(all)
```

```
##   PassengerId      Survived          Pclass          Name
##  Min.   :   1   Min.   :0.0000   Min.   :1.000   Length:1309
##  1st Qu.: 328   1st Qu.:0.0000   1st Qu.:2.000   Class :character
##  Median : 655   Median :0.0000   Median :3.000   Mode  :character
##  Mean   : 655   Mean   :0.2613   Mean   :2.295
##  3rd Qu.: 982   3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :1309   Max.   :1.0000   Max.   :3.000
##
##      Sex                 Age            SibSp            Parch
##  Length:1309        Min.   : 0.17   Min.   :0.0000   Min.   :0.000
##  Class :character   1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.000
##  Mode  :character   Median :28.00   Median :0.0000   Median :0.000
##                     Mean   :29.88   Mean   :0.4989   Mean   :0.385
##                     3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.000
##                     Max.   :80.00   Max.   :8.0000   Max.   :9.000
##                     NA's   :263
##     Ticket               Fare             Cabin
##  Length:1309        Min.   :  0.000   Length:1309
##  Class :character   1st Qu.:  7.896   Class :character
##  Mode  :character   Median : 14.454   Mode  :character
##                     Mean   : 33.295
##                     3rd Qu.: 31.275
##                     Max.   :512.329
##                     NA's   :1
##    Embarked
##  Length:1309
##  Class :character
##  Mode  :character
```

```
##
##
##
##
```

In the cell below, we transform the Sex and Embarked variables to factors.

```r
all$Sex <- factor(all$Sex)
all$Embarked <- factor(all$Embarked, exclude="")

summary(all)
```

```
##   PassengerId      Survived          Pclass          Name          
##  Min.   :   1   Min.   :0.0000   Min.   :1.000   Length:1309       
##  1st Qu.: 328   1st Qu.:0.0000   1st Qu.:2.000   Class :character  
##  Median : 655   Median :0.0000   Median :3.000   Mode  :character  
##  Mean   : 655   Mean   :0.2613   Mean   :2.295                     
##  3rd Qu.: 982   3rd Qu.:1.0000   3rd Qu.:3.000                     
##  Max.   :1309   Max.   :1.0000   Max.   :3.000                     
##                                                                    
##      Sex           Age            SibSp            Parch      
##  female:466   Min.   : 0.17   Min.   :0.0000   Min.   :0.000  
##  male  :843   1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.000  
##               Median :28.00   Median :0.0000   Median :0.000  
##               Mean   :29.88   Mean   :0.4989   Mean   :0.385  
##               3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.000  
##               Max.   :80.00   Max.   :8.0000   Max.   :9.000  
##               NA's   :263                                     
##     Ticket               Fare             Cabin            Embarked  
##  Length:1309        Min.   :  0.000   Length:1309        C   :270  
##  Class :character   1st Qu.:  7.896   Class :character   Q   :123  
##  Mode  :character   Median : 14.454   Mode  :character   S   :914  
##                     Mean   : 33.295                      NA's:  2  
##                     3rd Qu.: 31.275                                
##                     Max.   :512.329                                
##                     NA's   :1
```

Summary of the missing data

```r
sapply(all, function(attribute) {sum(is.na(attribute)==TRUE)/ length(attribute)
;})
```

```
##  PassengerId     Survived       Pclass         Name          Sex 
## 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000 
##          Age        SibSp        Parch       Ticket         Fare 
## 0.2009167303 0.0000000000 0.0000000000 0.0000000000 0.0007639419 
##        Cabin     Embarked 
## 0.0000000000 0.0015278839
```

The output above shows that there are mising values for variables Age, Fare and Embarked. We addressed the missing data for Fare and Embarked in the following way:

- Assign missing Embarked data to the most counted port (‘S’).
- Replace the missing Fare data by the mean fare.

```r
all$Embarked[which(is.na(all$Embarked))] <- 'S'
all$Fare[which(is.na(all$Fare))] <- mean(all$Fare, na.rm=TRUE)
```

```r
summary(all)
```

```
##   PassengerId      Survived          Pclass          Name          
##  Min.   :   1   Min.   :0.0000   Min.   :1.000   Length:1309       
##  1st Qu.: 328   1st Qu.:0.0000   1st Qu.:2.000   Class :character  
##  Median : 655   Median :0.0000   Median :3.000   Mode  :character  
##  Mean   : 655   Mean   :0.2613   Mean   :2.295                     
##  3rd Qu.: 982   3rd Qu.:1.0000   3rd Qu.:3.000                     
##  Max.   :1309   Max.   :1.0000   Max.   :3.000                     
##                                                                    
##      Sex           Age            SibSp            Parch      
##  female:466   Min.   : 0.17   Min.   :0.0000   Min.   :0.000  
##  male  :843   1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.000  
##               Median :28.00   Median :0.0000   Median :0.000  
##               Mean   :29.88   Mean   :0.4989   Mean   :0.385  
##               3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.000  
##               Max.   :80.00   Max.   :8.0000   Max.   :9.000  
##               NA's   :263                                     
##     Ticket               Fare            Cabin             Embarked
##  Length:1309        Min.   :  0.000   Length:1309        C:270   
##  Class :character   1st Qu.:  7.896   Class :character   Q:123   
##  Mode  :character   Median : 14.454   Mode  :character   S:916   
##                     Mean   : 33.295                              
##                     3rd Qu.: 31.275                              
##                     Max.   :512.329                              
## 
```

The Cabin column data is managed as character data. However there are many empty strings. Moreover this variable doesn't provide any relevant information. Therefore we will not use this feature for the modeling part.

```r
sum(all$Cabin == "")/nrow(all)
```

```
## [1] 0.7746371
```

There are 77% of empty strings for the Cabin column.

Age Missing Data imputation

1. Check the title frequency

```r
table_words = table(unlist(strsplit(all$Name, "\\s+")))
sort(table_words [grep('\\.',names(table_words))], decreasing=TRUE)
```

```
## 
##       Mr.     Miss.      Mrs.   Master.       Dr.      Rev.      Col. 
##       757       260       197        61         8         8         4 
##    Major.     Mlle.       Ms.     Capt. Countess.      Don.     Dona. 
##         2         2         2         1         1         1         1 
## Jonkheer.        L.     Lady.      Mme.      Sir. 
##         1         1         1         1         1
```

2. Find missing age by title

```r
library(stringr)
tb_data = cbind(all$Age, str_match(all$Name, " [a-zA-Z]+\\."))
table(tb_data[is.na(tb_data[,1]),2])
```

```
##
##      Dr.  Master.    Miss.      Mr.     Mrs.      Ms.
##        1        8       50      176       27        1
```

3. Compute mean value by titles

```r
mean.mr = mean(all$Age[grepl(" Mr\\.", all$Name)],na.rm=TRUE)
mean.mrs = mean(all$Age[grepl(" Mrs\\.", all$Name)],na.rm=TRUE)
mean.dr = mean(all$Age[grepl(" Dr\\.", all$Name)],na.rm=TRUE)
mean.miss = mean(all$Age[grepl(" Miss\\.", all$Name)],na.rm=TRUE)
mean.master =  mean(all$Age[grepl(" Master\\.", all$Name)],na.rm=TRUE)
```

4. Apply the mean to the missing data

```r
all$Age[grepl(" Mr\\.", all$Name)
                & is.na(all$Age)] = mean.mr
all$Age[grepl(" Mrs\\.", all$Name)
                & is.na(all$Age)] = mean.mrs
all$Age[grepl(" Dr\\.", all$Name)
                & is.na(all$Age)] = mean.dr
all$Age[grepl(" Miss\\.", all$Name)
                & is.na(all$Age)] = mean.miss
all$Age[grepl(" Master\\.", all$Name)
                & is.na(all$Age)] = mean.master
# Special case for Ms. that we manage as Miss.
all$Age[grepl(" Ms\\.", all$Name)
                & is.na(all$Age)] = mean.miss
```

5. Check that there is no remaining missing age values

```r
sum(is.na(all$Age) == TRUE) /  length(all$Age)
```

```
## [1] 0
```

Data transformation

Manage class and sex as factors instead of numbers

```r
all$Pclass <- factor(all$Pclass)
all$Sex <- factor(all$Sex)
#all$Embarked <- factor(all$Embarked)
```

Add a variable title

```r
all$Title <- substring(str_extract(all$Name, '\\, \\w*\\.'), 3)
all$Title <- factor(all$Title)
#all$Title <- gsub('([[:alpha:]]*\\, )([[:alpha:]]*\\.)([[:alpha:]]*)','\\2',all$Name)

#all$Title <- str_replace(all$Name, '(\\w*)\\, (\\w*\\.)(\\w*)',REF2)
```

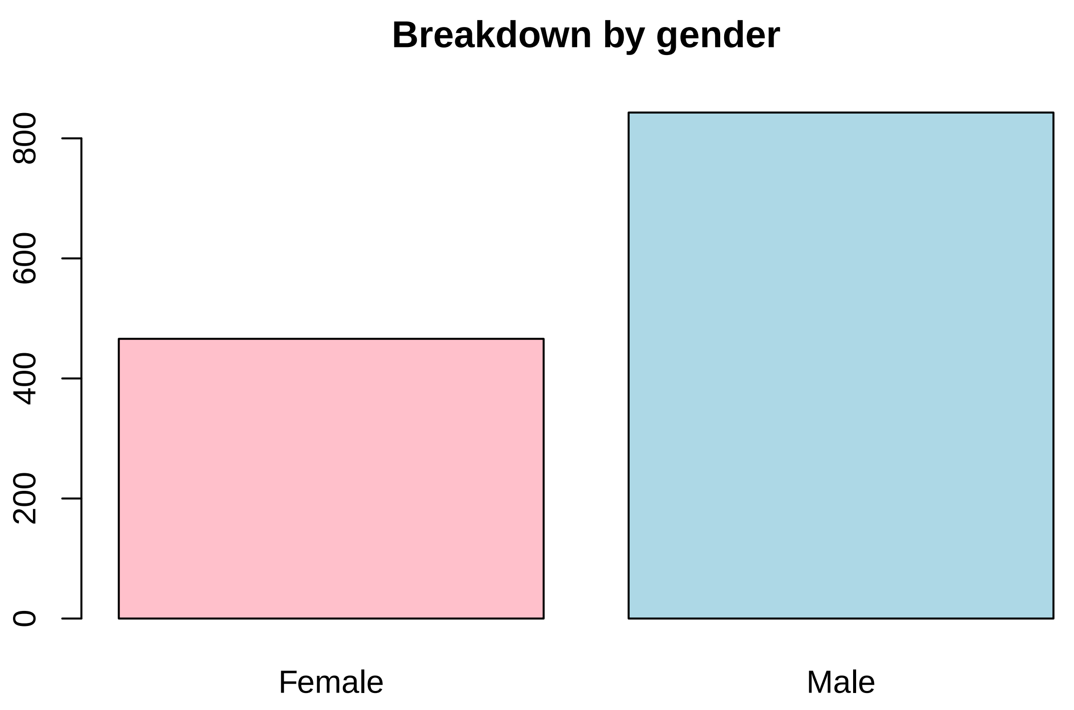
Exploratory data analysis

Breakdown by gender

```r
table(all$Sex)
```

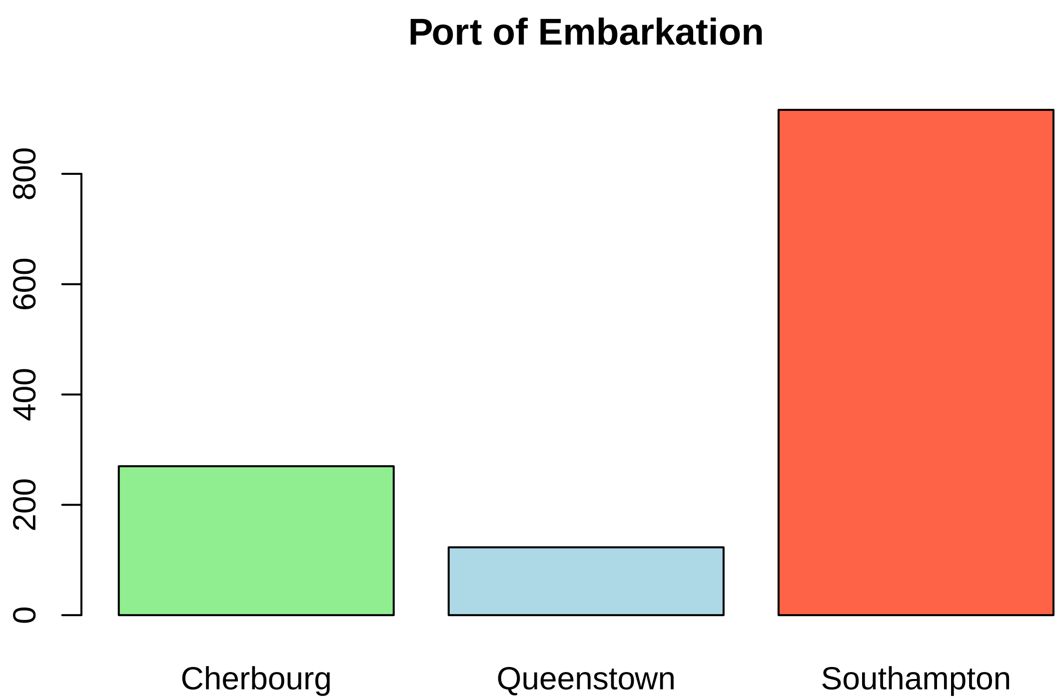
```
##
## female   male
##    466    843
```

```
barplot(table(all$Sex), names= c("Female","Male"), col= c("pink", "lightblue"), main="Breakdown by gende
```



Breakdown by port of Embarkation

```
barplot(table(all$Embarked), col=c("lightgreen","lightblue","tomato"), names= c("Cherbourg", "Queenstow
```

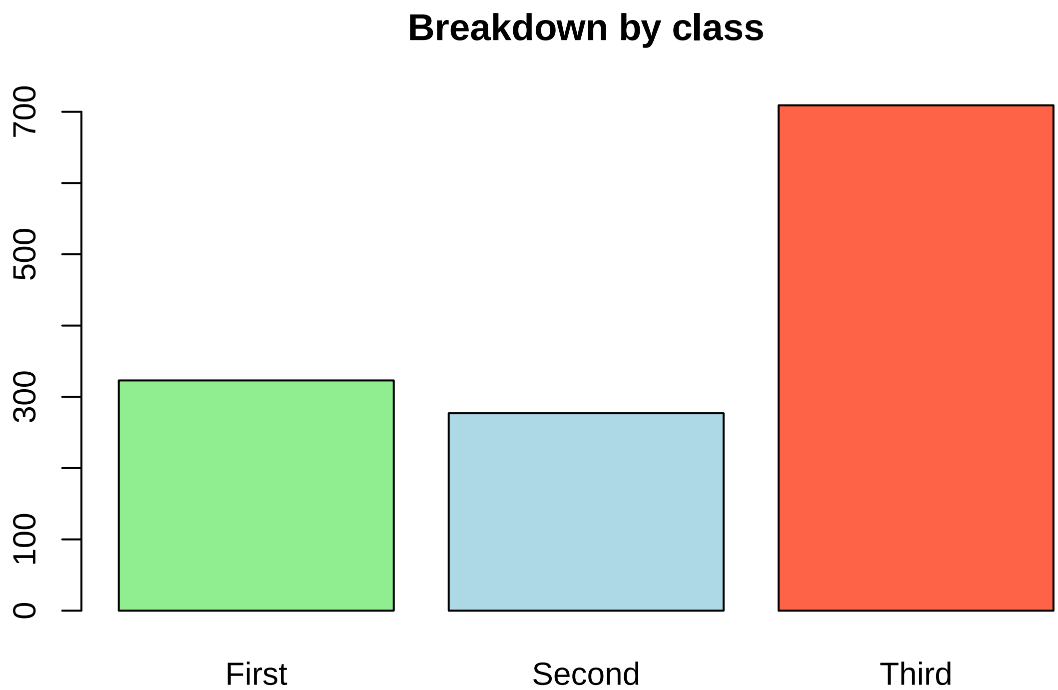


Breakdown by class

```
table(all$Pclass)
```

```
##
##   1   2   3
## 323 277 709
```

```
barplot(table(all$Pclass), col=c("lightgreen", "lightblue", "tomato"),
        names= c("First", "Second", "Third"), main="Breakdown by class")
```

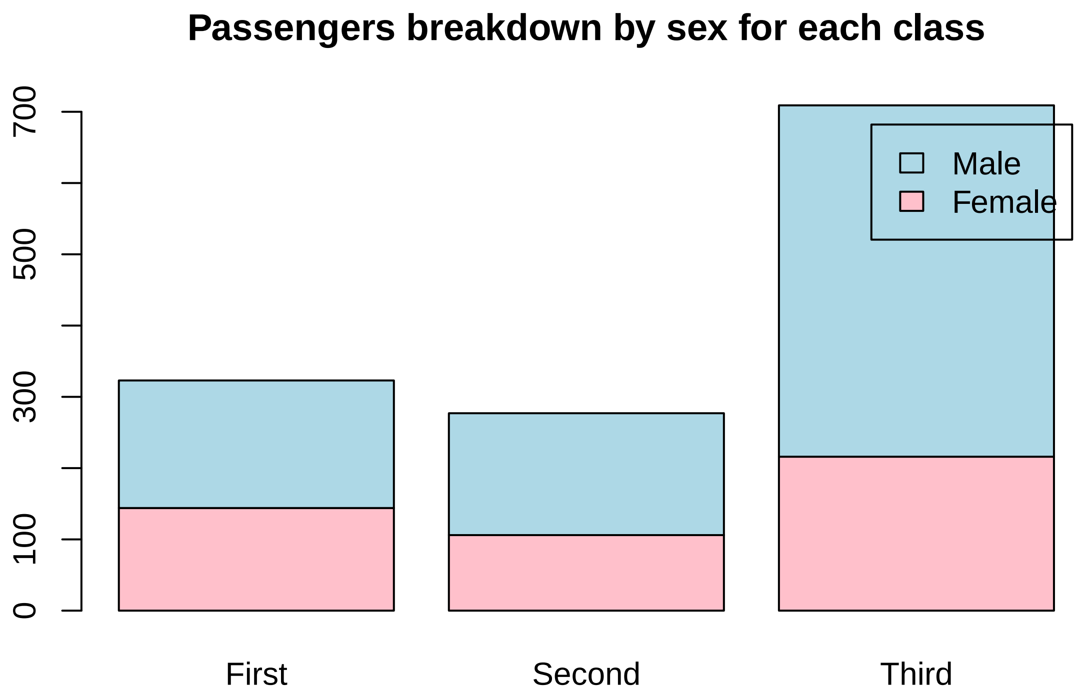


Breakdown by sex for each class

```
table(all$Sex, all$Pclass )
```

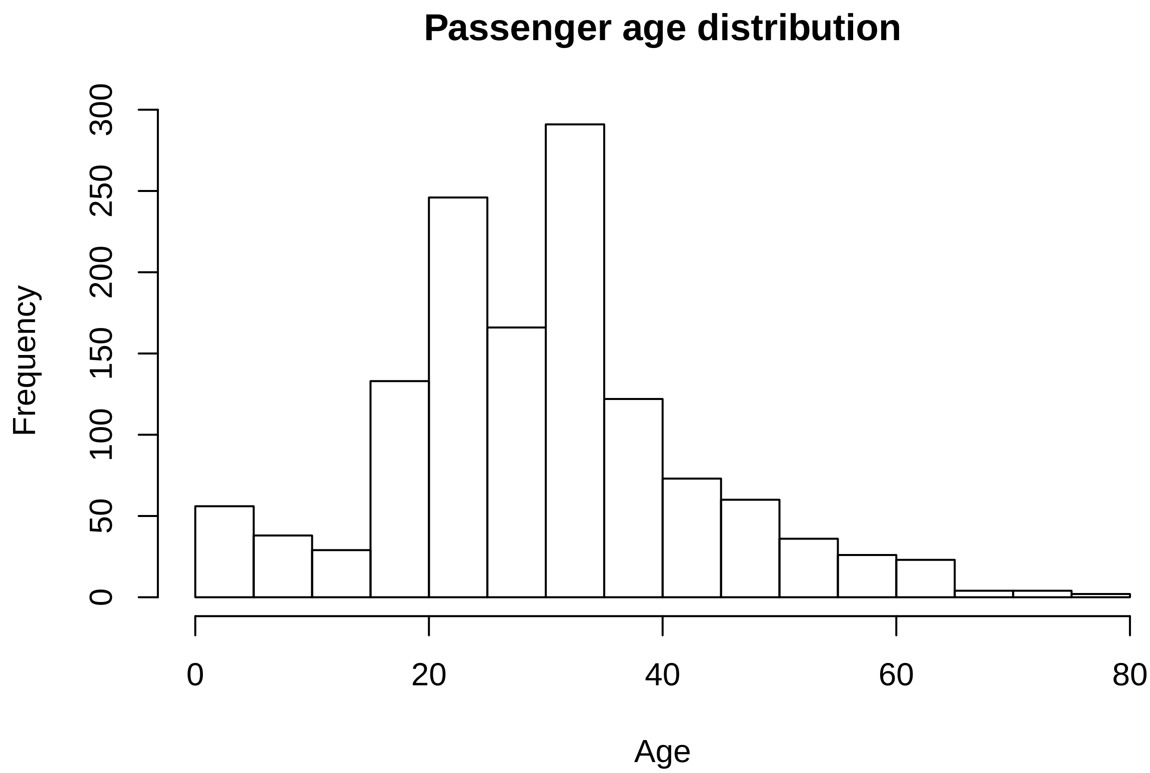
```
##
##            1   2   3
##   female 144 106 216
##   male   179 171 493
```

```
countsTable <- table(all$Sex, all$Pclass )
barplot(countsTable, col=c("pink", "lightblue"), legend=c("Female", "Male"),
        names=c("First", "Second", "Third"), main="Passengers breakdown by sex for each class")
```



Hist distribution by passenger age

```
hist(all$Age, main="Passenger age distribution", xlab="Age")
```

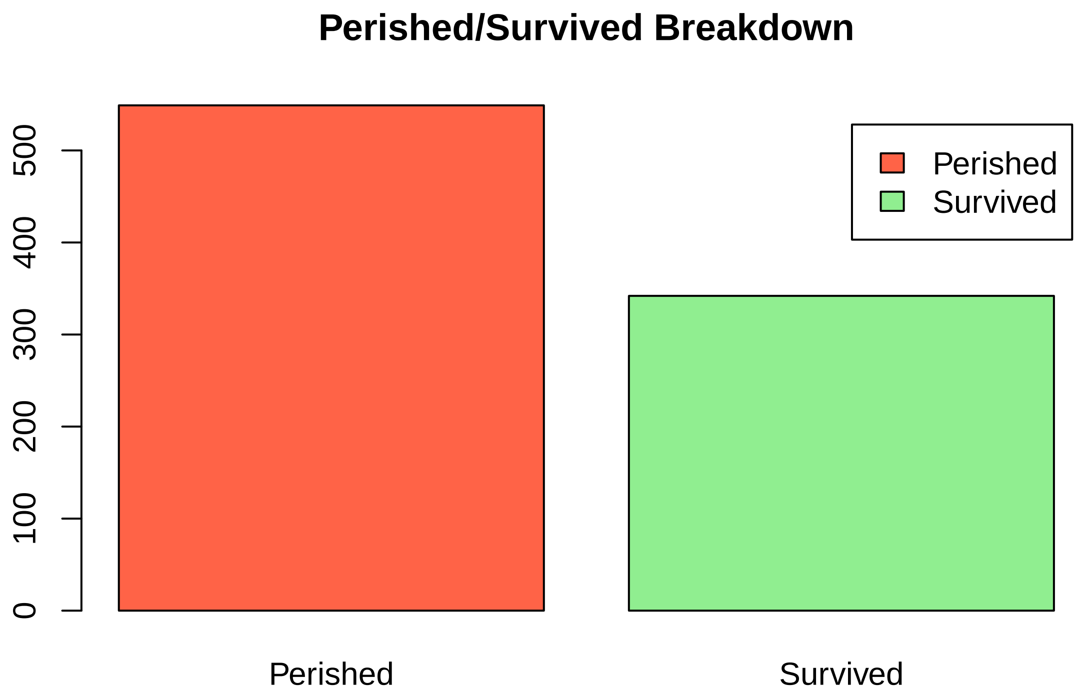


Spliting back the data in train and test sets

```
dt <- 1:nrow(train)
train <- all[dt,]
#mrow_all <- nrow(all)
test <- all[-dt,]
test$Survived <- NULL
```

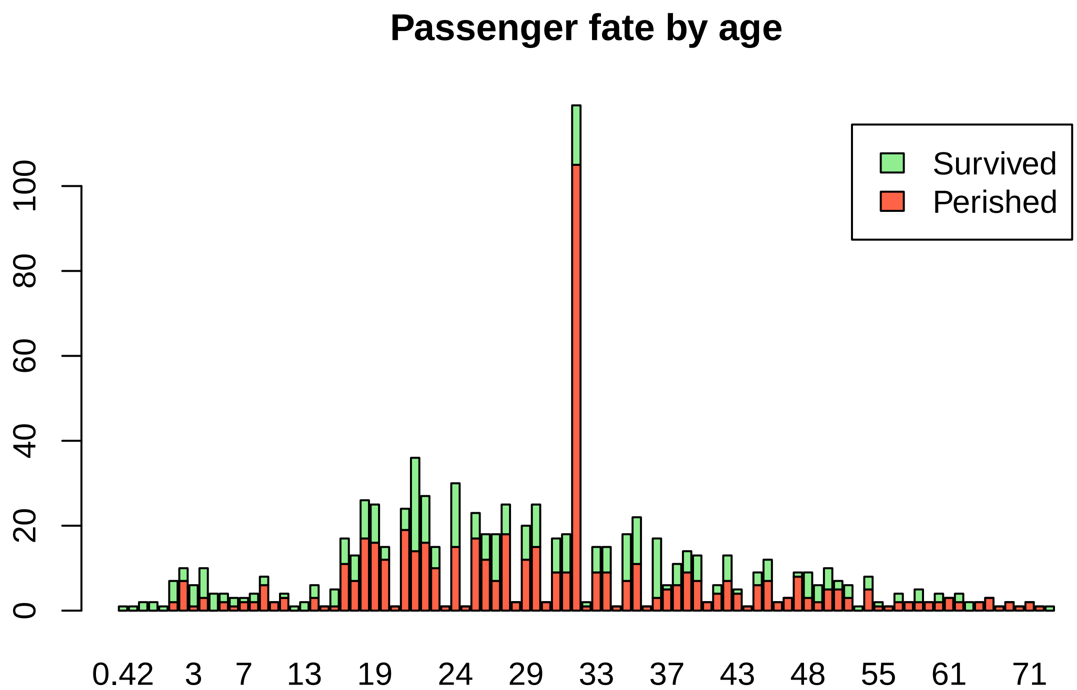
Specific Exploratory analysis of the training data set

```
barplot(table(train$Survived), col=c("Tomato", "lightgreen"),
        names=c("Perished", "Survived"), legend=c("Perished", "Survived"),
        main="Perished/Survived Breakdown" )
```



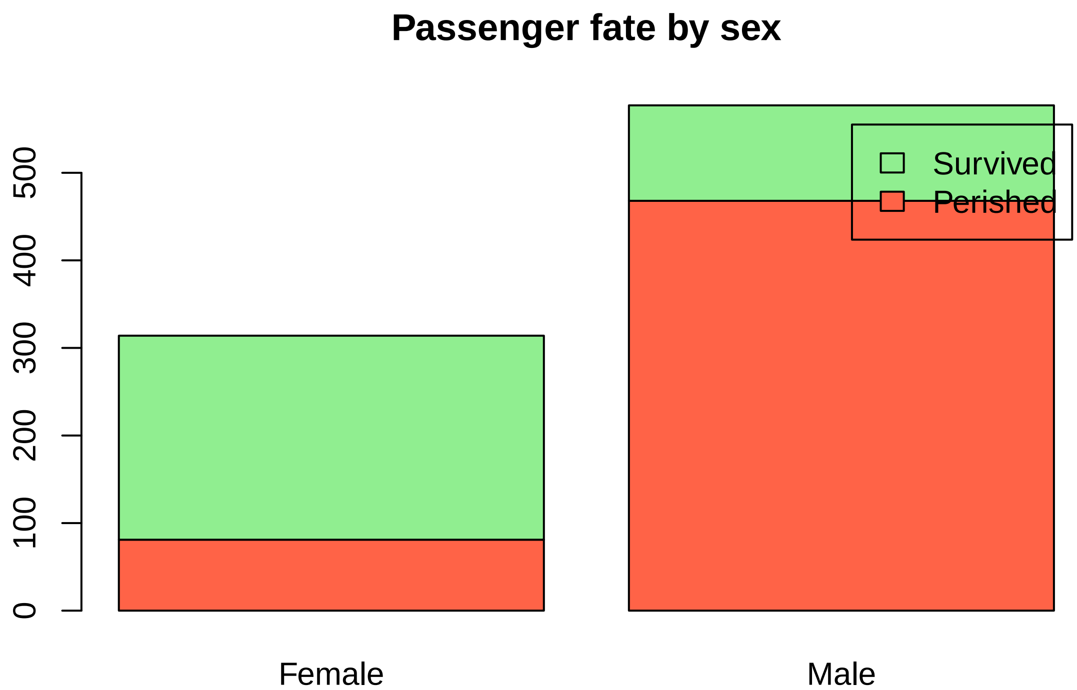
Passenger fate by age

```
barplot(table(train$Survived, train$Age), col=c("Tomato", "lightgreen"),
       legend=c("Perished", "Survived"),
        main="Passenger fate by age" )
```



Passenger fate by sex

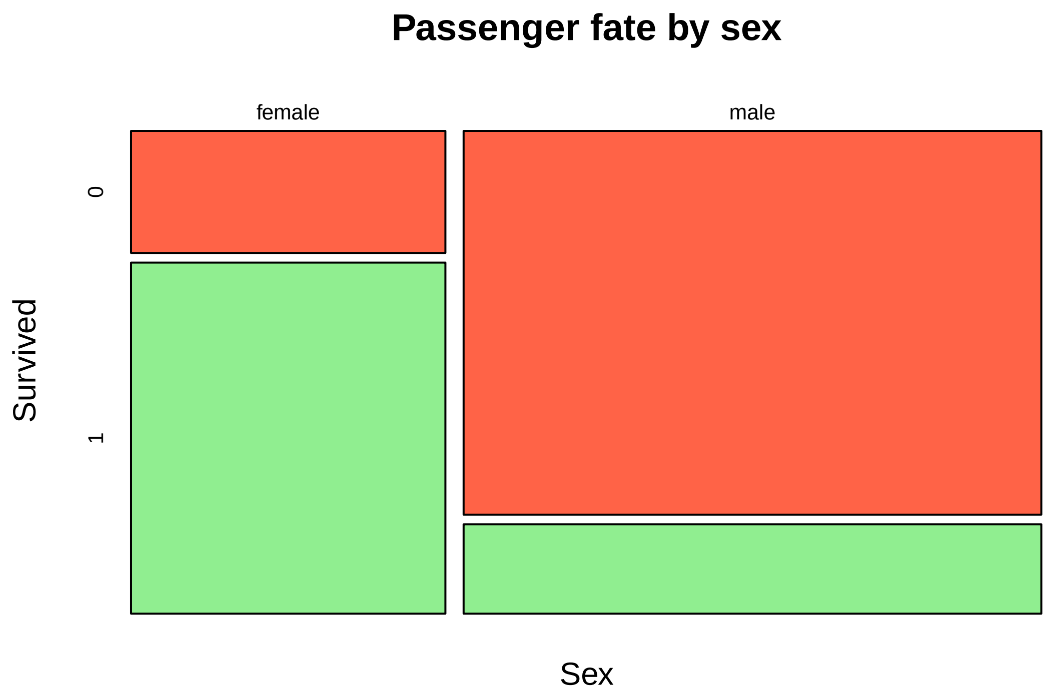
```
barplot(table(train$Survived, train$Sex), col=c("Tomato", "lightgreen"),
       names=c("Female", "Male"), legend=c("Perished", "Survived"),
        main="Passenger fate by sex" )
```



Mosaic plot of the same data

```
mosaicplot( train$Sex~train$Survived, main="Passenger fate by sex",
            SHADE=FALSE, col=c("Tomato", "lightgreen"),xlab="Sex", ylab="Survived")
```

```
## Warning: In mosaicplot.default(table(mf), main = main, ...) :
##  extra argument 'SHADE' will be disregarded
```



Passenger fate by travelling class

```
table(train$Survived)
```

```
##
##   0   1
## 549 342
```

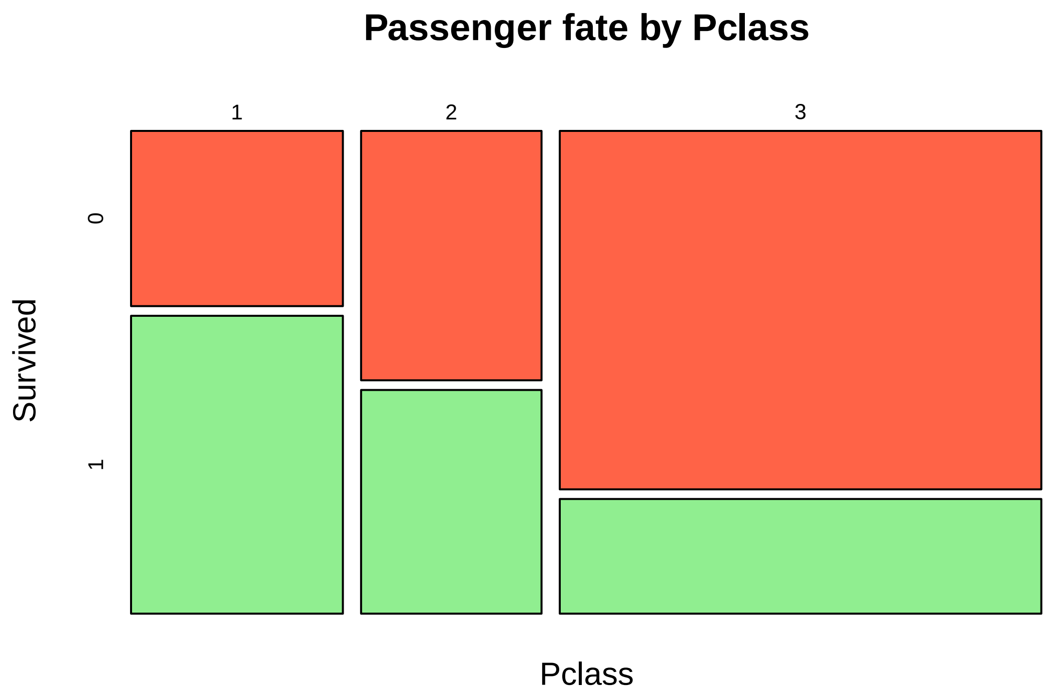
```
table(train$Survived, train$Pclass)
```

```
##
##       1   2   3
##   0  80  97 372
##   1 136  87 119
```

```
barplot( table(train$Survived, train$Pclass), col=c("Tomato", "lightgreen"),
        legend = c("Perished", "Survived"), names= c("First", "Second", "Third"),
        main= "Passenger fate by Class" )
```

Corresponding Mosaic Plot

```
mosaicplot(train$Pclass ~ train$Survived, main="Passenger fate by Pclass",
           shade=FALSE, color=c("tomato","lightgreen"), xlab="Pclass", ylab="Survived")
```



Predicting passenger survival using Decision Tree

For the modelling part, we will not take into account the following variables:

- PassengerId: This is just an identifier for the passenger and doesn't bring any value.
- Ticket: This is just the ticket number and this doesn't add also any value
- cabin: This is a cabin identification number that is not relevant for this analysis

```
library(rpart)
# Step 1: Build the maximal tree

Tree <- rpart(Survived~Pclass + Sex + Age + SibSp + Parch + Fare + Embarked, data=train,
              method="class", control=rpart.control(minsplit=2,cp=0))

#Tree
```

Summary r include=FALSE

```
#summary(Tree)
```

Error on the maximal tree

```
pred <- predict(Tree, type="class")
error<-  1/length(train$Survived) * sum(train$Survived != pred )
error
```

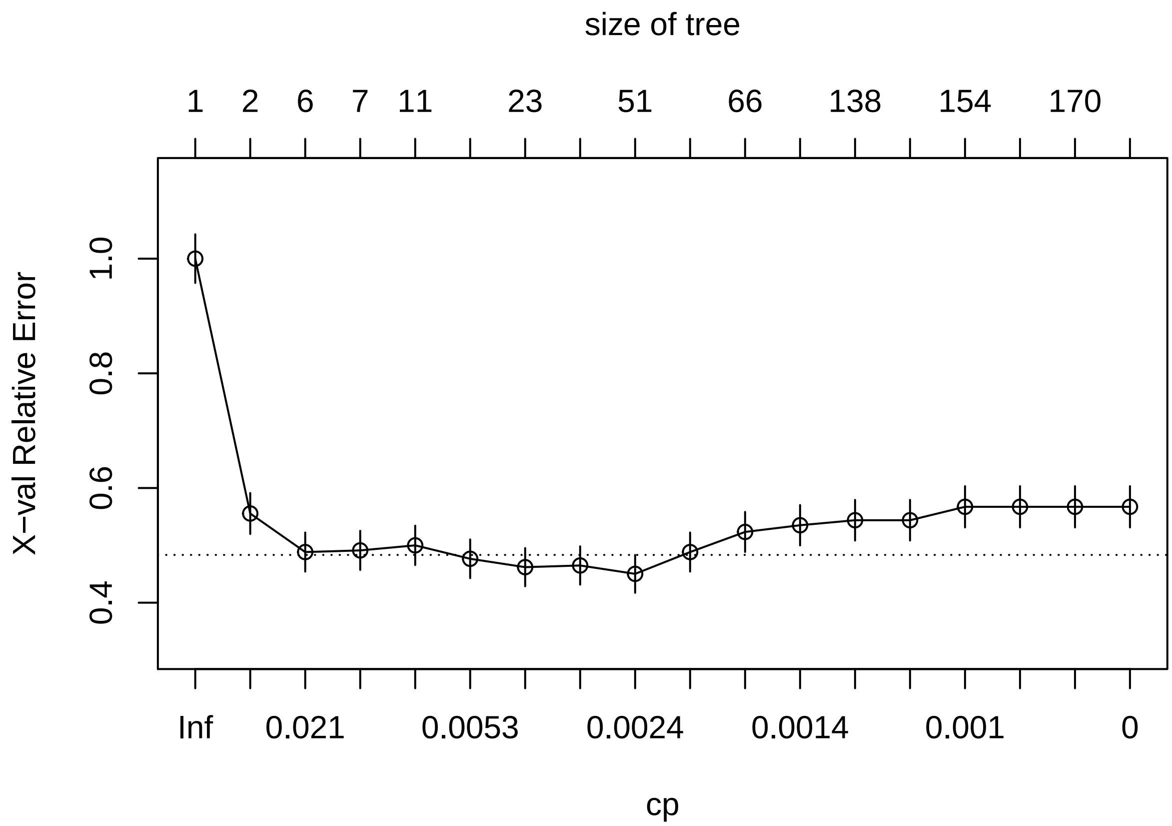
```
## [1] 0.01795735
```

printcp

```r
A <- printcp(Tree)
```

```
##
## Classification tree:
## rpart(formula = Survived ~ Pclass + Sex + Age + SibSp + Parch +
##     Fare + Embarked, data = train, method = "class", control = rpart.control(minsplit = 2,
##     cp = 0))
##
## Variables actually used in tree construction:
## [1] Age      Embarked Fare     Parch    Pclass   Sex      SibSp
##
## Root node error: 342/891 = 0.38384
##
## n= 891
##
##            CP nsplit rel error  xerror     xstd
## 1  0.44444444      0  1.000000 1.00000 0.042446
## 2  0.03070175      1  0.555556 0.55556 0.035750
## 3  0.01461988      5  0.432749 0.48830 0.034061
## 4  0.00730994      6  0.418129 0.49123 0.034140
## 5  0.00584795     10  0.383041 0.50000 0.034372
## 6  0.00487329     12  0.371345 0.47661 0.033744
## 7  0.00438596     22  0.312865 0.46199 0.033336
## 8  0.00292398     29  0.280702 0.46491 0.033419
## 9  0.00194932     50  0.219298 0.45029 0.033001
## 10 0.00167084     56  0.207602 0.48830 0.034061
## 11 0.00146199     65  0.190058 0.52339 0.034970
## 12 0.00125313    122  0.105263 0.53509 0.035260
## 13 0.00116959    137  0.084795 0.54386 0.035472
## 14 0.00109649    142  0.078947 0.54386 0.035472
## 15 0.00097466    153  0.064327 0.56725 0.036021
## 16 0.00073099    159  0.058480 0.56725 0.036021
## 17 0.00058480    169  0.049708 0.56725 0.036021
## 18 0.00000000    174  0.046784 0.56725 0.036021
```

```r
plotcp(Tree)
```



Step 2: Pruning

```
mincp <- which(A[,4] == min(A[,4]))
mincp
```

```
## 9
## 9
```

```
#cpthres: 1-SE rule threshold : Error_min + standard_error
cpthres <- A[mincp,4] + A[mincp,5]
cp1se <- min(which(A[,4] <= cpthres))
#cp1se <- which(min(A[cand,4]) == A[,4])
cp1se
```

```
## [1] 6
```

The lower xerror is 0.44 with a standard error of 0.03. When we apply the 1SE rule, we get 0.45 + 0.03 = 0.47 as threshold. Therefore the final tree is the smaller tree (less splits)one with error lower than 0.47 It's the value corresponding to cp=0.00487329 (Id = 6). The code above gives an accurate way to identify the cp corresponding to the 1SE cp rule.

Alternative method

```
cverr=A[,4]
mincverr=which(cverr==min(cverr))
s=A[mincverr,4]+A[mincverr,5]
s=min(s)
B=1*(cverr<=s)
a=min(which(B==1))
```

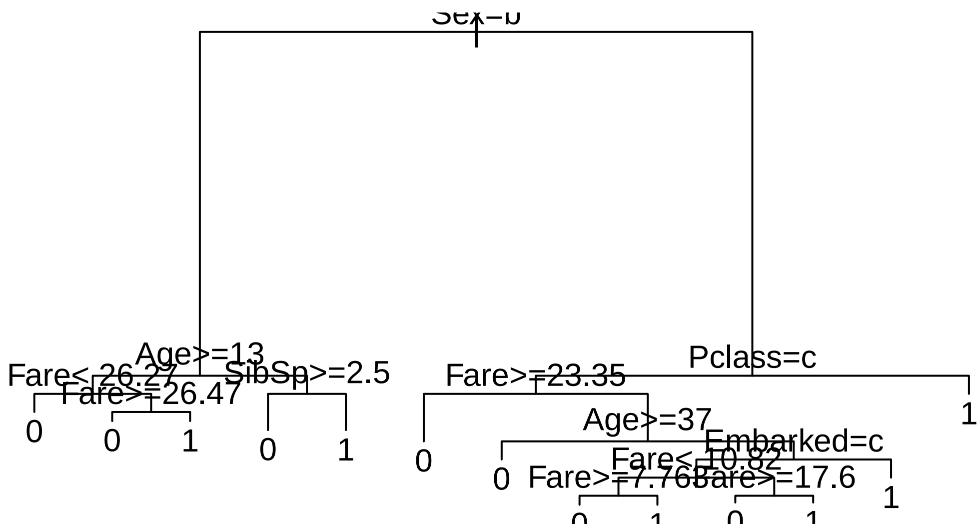
```r
a
```

```
## [1] 6
```

```r
cp=A[a,1]
cp
```

```
## [1] 0.004873294
```

```r
#Treep <- prune(Tree, cp=A[5,1])
Treep <- prune(Tree, cp=A[cp1se,1])
plot(Treep)
text(Treep)
```



Display a more fancy plot

Install package and load library

```r
#install.packages('rattle')
#install.packages('rpart.plot')
#install.packages('RColorBrewer')
#library(rattle)
```

```r
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.6.2
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.3.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

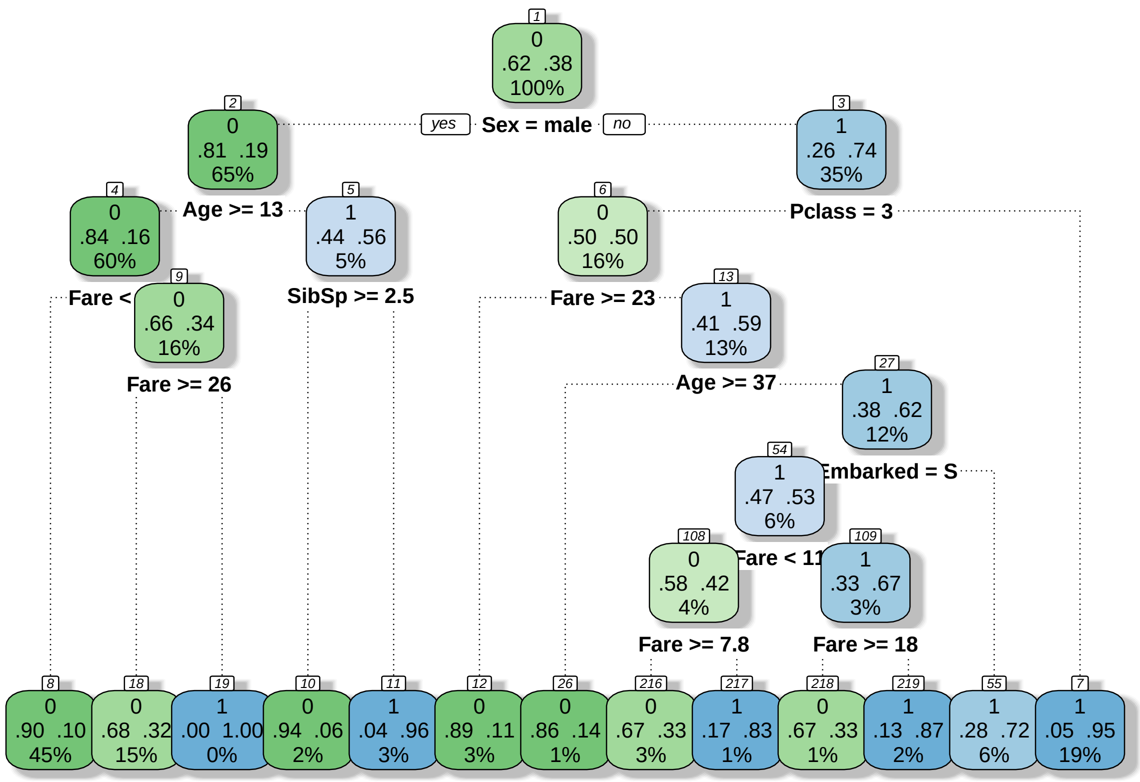
```
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.6.2
```

```
library(RColorBrewer)
```

Tree plot

```
fancyRpartPlot(Treep, tweak=1.4)
```



Rattle 2020−Mar−28 15:06:57 jose

We see from the tree, that the most important variables are respectively: Sex, Age, Pclass and SibSp.

Prediction on the test set

```
pred_dt <- predict(Treep, newdata=test, type="class")
submit_dt <- data.frame(PassengerId = test$PassengerId, Survived = pred_dt)
```

Write submission

```
write.csv(submit_dt, file = "submit_dt_02.csv", row.names = FALSE)
```

After submission, Kaggle score is 0.79425.

Random Forest

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.6.1
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:rattle':
##
##     importance
```

```
# Set seed for reproducibility
set.seed(1234)
Tree_rf <- randomForest(as.factor(Survived) ~ Pclass + Sex + Age + SibSp + Parch +
    Fare + Embarked, data=train, importance=TRUE, proximity=TRUE, ntree=1000)

Tree_rf
```

```
##
## Call:
##  randomForest(formula = as.factor(Survived) ~ Pclass + Sex + Age +      SibSp + Parch + Fare + Embar
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 17.28%
## Confusion matrix:
##     0   1 class.error
## 0 503  46  0.08378871
## 1 108 234  0.31578947
```

Prediction

```
pred_rf <- predict(Tree_rf, newdata=test, type="class")
submit_rf <- data.frame(PassengerId = test$PassengerId, Survived = pred_rf)
```

Write submission

```
write.csv(submit_rf, file = "submit_rf2.csv", row.names = FALSE)
```

The score for Random Forrest is 0.77990 and therefore worse that what we get for decision tree.

Using CART without prior missing data inputation

We assume here that the train_raw data.frame is the raw data.frame with missing values:

```
summary(train_raw)
```

```
##   PassengerId       Survived        Pclass       Name               Sex
##  Min.   :  1.0   Min.   :0.0000   1:216   Length:891         female:314
##  1st Qu.:223.5   1st Qu.:0.0000   2:184   Class :character   male  :577
##  Median :446.0   Median :0.0000   3:491   Mode  :character
##  Mean   :446.0   Mean   :0.3838
##  3rd Qu.:668.5   3rd Qu.:1.0000
##  Max.   :891.0   Max.   :1.0000
##
##       Age            SibSp           Parch           Ticket
##  Min.   : 0.42   Min.   :0.000   Min.   :0.0000   Length:891
##  1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000   Class :character
##  Median :28.00   Median :0.000   Median :0.0000   Mode  :character
##  Mean   :29.70   Mean   :0.523   Mean   :0.3816
```

```
##  3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##  Max.   :80.00   Max.   :8.000   Max.   :6.0000
##  NA's   :177
##       Fare            Cabin            Embarked
##  Min.   :  0.00   Length:891         C   :168
##  1st Qu.:  7.91   Class :character   Q   : 77
##  Median : 14.45   Mode  :character   S   :644
##  Mean   : 32.20                      NA's:  2
##  3rd Qu.: 31.00
##  Max.   :512.33
##
```

We confirm that we still have missing values for Age and Embarked fields.

1. Build the maximal tree

```r
library(rpart)
# Step 1: Build the maximal tree

Tree_na <- rpart(Survived~Pclass + Sex + Age + SibSp + Parch + Fare + Embarked, data=train_raw,
              method="class", control=rpart.control(minsplit=2,cp=0))

#Tree_na
```

printcp

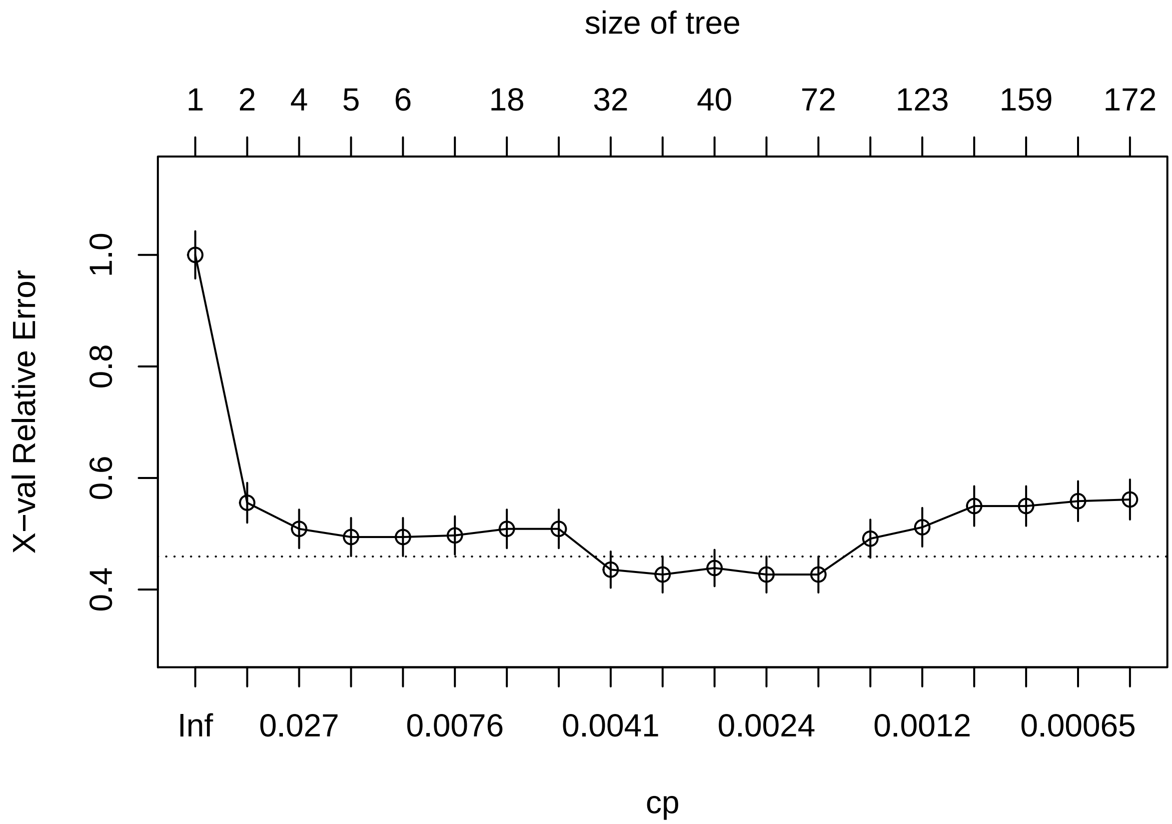
```r
A_na <- printcp(Tree_na)
```

```
##
## Classification tree:
## rpart(formula = Survived ~ Pclass + Sex + Age + SibSp + Parch +
##     Fare + Embarked, data = train_raw, method = "class", control = rpart.control(minsplit = 2,
##     cp = 0))
##
## Variables actually used in tree construction:
## [1] Age      Embarked Fare     Parch    Pclass   Sex      SibSp
##
## Root node error: 342/891 = 0.38384
##
## n= 891
##
##            CP nsplit rel error  xerror     xstd
## 1  0.44444444      0  1.000000 1.00000 0.042446
## 2  0.03070175      1  0.555556 0.55556 0.035750
## 3  0.02339181      3  0.494152 0.50877 0.034599
## 4  0.02046784      4  0.470760 0.49415 0.034217
## 5  0.00877193      5  0.450292 0.49415 0.034217
## 6  0.00657895     10  0.403509 0.49708 0.034295
## 7  0.00584795     17  0.356725 0.50877 0.034599
## 8  0.00438596     18  0.350877 0.50877 0.034599
## 9  0.00389864     31  0.292398 0.43567 0.032571
## 10 0.00350877     34  0.280702 0.42690 0.032306
## 11 0.00292398     39  0.263158 0.43860 0.032658
## 12 0.00194932     68  0.178363 0.42690 0.032306
## 13 0.00146199     71  0.172515 0.42690 0.032306
## 14 0.00132908    107  0.119883 0.49123 0.034140
```

```
## 15 0.00116959    122  0.099415 0.51170 0.034675
## 16 0.00097466    136  0.078947 0.54971 0.035612
## 17 0.00073099    158  0.055556 0.54971 0.035612
## 18 0.00058480    166  0.049708 0.55848 0.035818
## 19 0.00000000    171  0.046784 0.56140 0.035886
```

plotcp

```
plotcp(Tree_na)
```

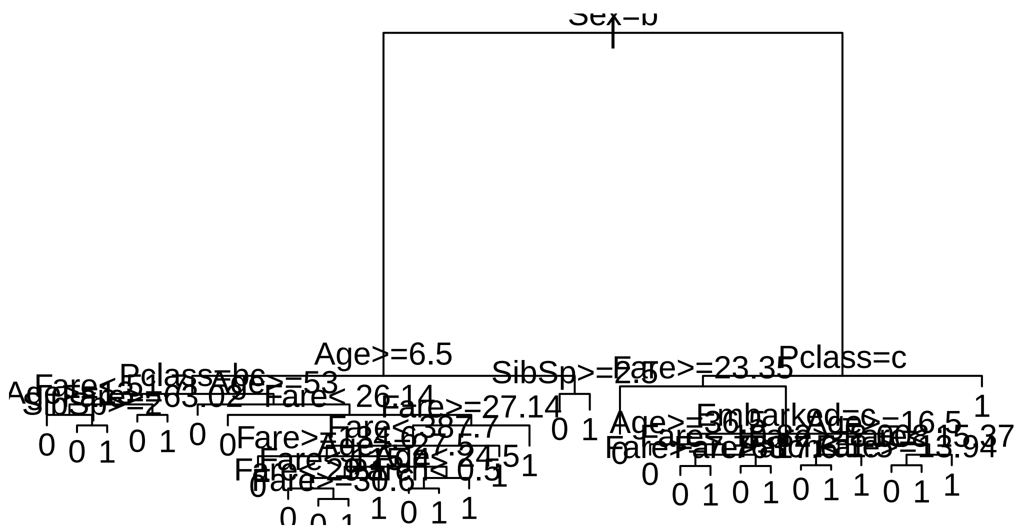



```
mincp <- which(A_na[,4] == min(A_na[,4]))
# as there are several min
mincps <- min(mincp)
#cpthres: 1-SE rule threshold : Error_min + standard_error
cpthres <- A_na[mincps,4] + A_na[mincps,5]
cp1se <- min(which(A_na[,4] <= cpthres))
#cp1se <- which(min(A[cand,4]) == A[,4])
cp1se
```

```
## [1] 9
```

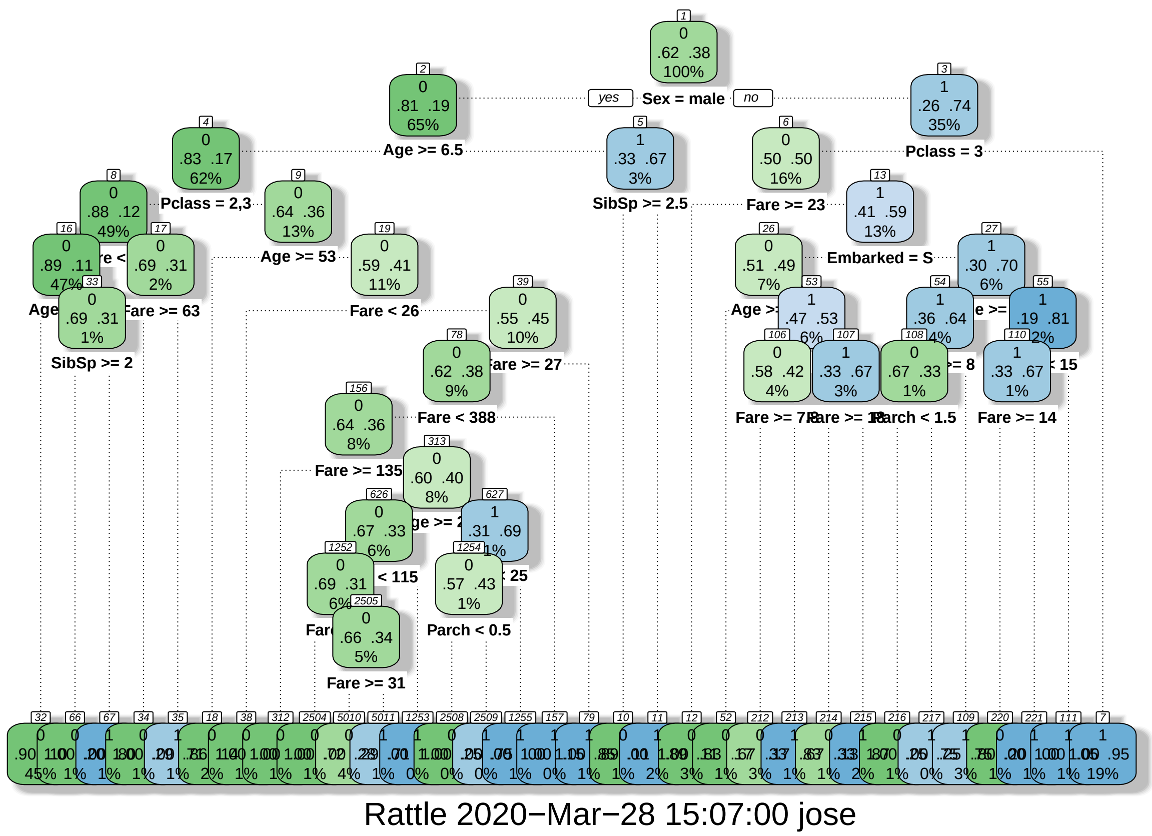
2. Pruning

```
Tree_nap <- prune(Tree_na, cp=A_na[cp1se,1])
plot(Tree_nap)
text(Tree_nap)
```



Fancy tree plot

```
fancyRpartPlot(Tree_nap, tweak=2.5)
```



Prediction on the test set

```
pred_dt_na <- predict(Tree_nap, newdata=test_raw, type="class")
submit_dt_na <- data.frame(PassengerId = test_raw$PassengerId, Survived = pred_dt_na)
```

Write submission

```
write.csv(submit_dt_na, file = "submit_dt_na_02.csv", row.names = FALSE)
```

After submission, Kaggle score is 0.77511. Therefore it's worse than model with missing values inputation.

Using CART with the additional variable title

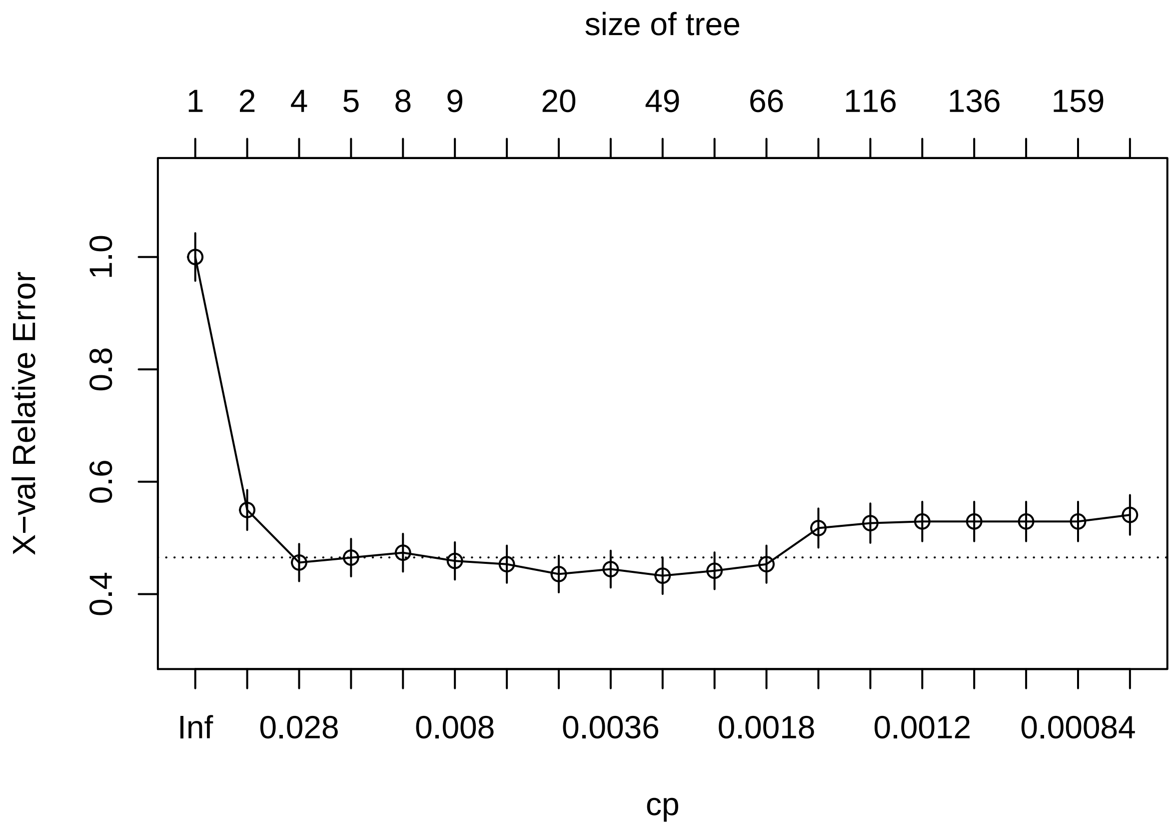
```
# Step 1: Build the maximal tree

Tree_title <- rpart(Survived~Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + Title, data=train,
              method="class", control=rpart.control(minsplit=2,cp=0))

#Tree
```

```
A_tl <- printcp(Tree_title)
```

```
##
## Classification tree:
## rpart(formula = Survived ~ Pclass + Sex + Age + SibSp + Parch +
##     Fare + Embarked + Title, data = train, method = "class",
##     control = rpart.control(minsplit = 2, cp = 0))
##
```

```
## Variables actually used in tree construction:
## [1] Age      Embarked Fare     Parch    Pclass   Sex      SibSp    Title
##
## Root node error: 342/891 = 0.38384
##
## n= 891
##
##            CP nsplit rel error  xerror     xstd
## 1  0.46198830      0  1.000000 1.00000 0.042446
## 2  0.05263158      1  0.538012 0.54971 0.035612
## 3  0.01461988      3  0.432749 0.45614 0.033170
## 4  0.00974659      4  0.418129 0.46491 0.033419
## 5  0.00877193      7  0.388889 0.47368 0.033663
## 6  0.00730994      8  0.380117 0.45906 0.033253
## 7  0.00584795     12  0.350877 0.45322 0.033086
## 8  0.00438596     19  0.309942 0.43567 0.032571
## 9  0.00292398     21  0.301170 0.44444 0.032831
## 10 0.00219298     48  0.219298 0.43275 0.032483
## 11 0.00194932     60  0.192982 0.44152 0.032745
## 12 0.00167084     65  0.181287 0.45322 0.033086
## 13 0.00146199     75  0.163743 0.51754 0.034823
## 14 0.00125313    115  0.102339 0.52632 0.035043
## 15 0.00116959    130  0.081871 0.52924 0.035116
## 16 0.00109649    135  0.076023 0.52924 0.035116
## 17 0.00097466    146  0.061404 0.52924 0.035116
## 18 0.00073099    158  0.049708 0.52924 0.035116
## 19 0.00000000    168  0.040936 0.54094 0.035402
```

```r
plotcp(Tree_title)
```

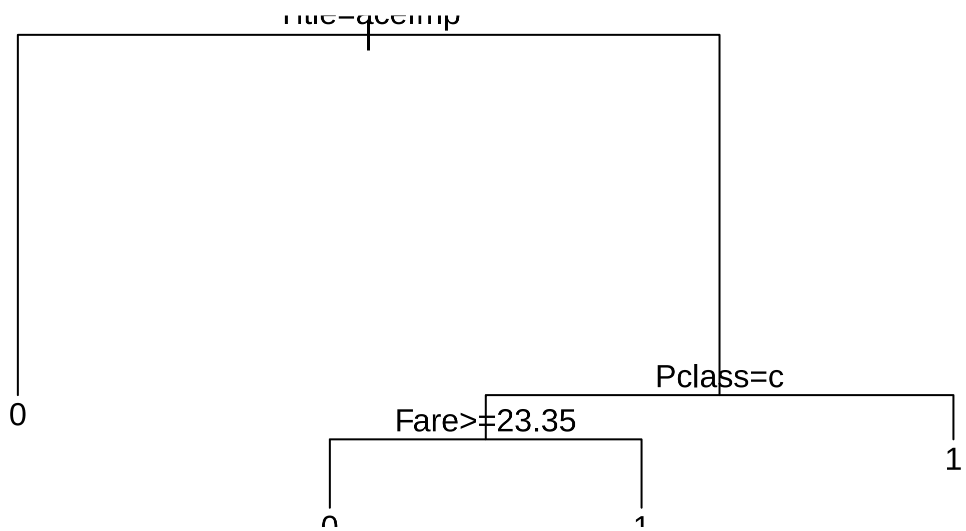



```
mincp <- which(A_tl[,4] == min(A_tl[,4]))
# as there are several min
mincps <- min(mincp)
#cpthres: 1-SE rule threshold : Error_min + standard_error
cpthres <- A_tl[mincps,4] + A_tl[mincps,5]
cp1se <- min(which(A_tl[,4] <= cpthres))
#cp1se <- which(min(A[cand,4]) == A[,4])
cp1se
```

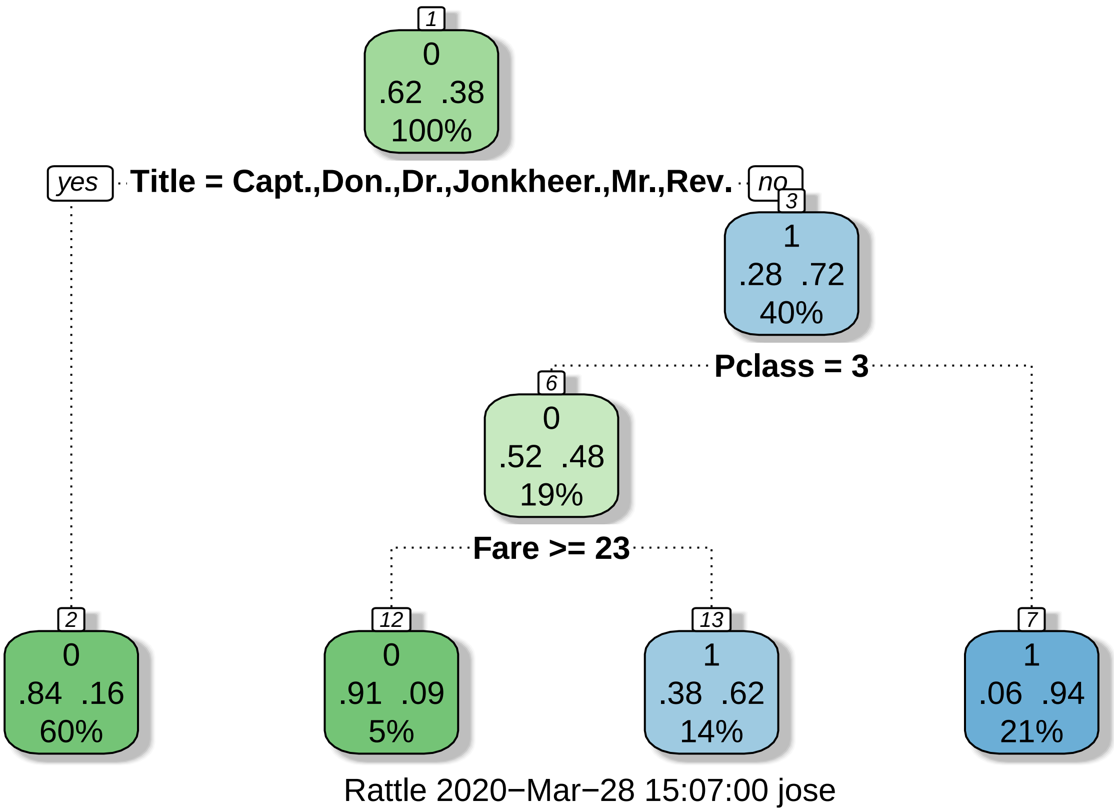
```
## [1] 3
```

Pruning

```
Tree_tlp <- prune(Tree_title, cp=A_tl[cp1se,1])
plot(Tree_tlp)
text(Tree_tlp)
```




```
fancyRpartPlot(Tree_tlp, tweak=1.0)
```



Prediction on the test set

```
# Add an extra factor Dona. for title Dona.
#levels(train$Title) <- c(levels(train$Title), "Dona.")
pred_dt_tl <- predict(Tree_tlp, newdata=test, type="class")
submit_dt_tl <- data.frame(PassengerId = test$PassengerId, Survived = pred_dt_tl)
```

Write submission

```
write.csv(submit_dt_tl, file = "submit_dt_tl_01.csv", row.names = FALSE)
```

After submission, Kaggle score is 0.79425

Conclusion

We tried to predict the fate of passengers in the test set using several CART and Random Forrest models. We got the best score with the CART model with simple inputations for missing data. For this specific problem we were not able to get better results with Random Forrest models.