**Use R to Predict the survival in Titanic disaster with machine learning techniques**

**Statistical Computing- Project 1**

**Background:** The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

**Objective:** In this project, I will complete the analysis of what sorts of people were likely to survive. In particular, I will apply the tools of machine learning to predict which passengers survived the tragedy.

**Dataset:** The data has been split into two groups:

* training set (train.csv)

The training set should be used to build your machine learning models. For the training set, we provide the outcome (also known as the “ground truth”) for each passenger. Your model will be based on “features” like passengers’ gender and class. You can also use feature engineering to create new features.

* test set (test.csv)

The test set should be used to see how well your model performs on unseen data. For the test set, we do not provide the ground truth for each passenger. It is your job to predict these outcomes. For each passenger in the test set, use the model you trained to predict whether or not they survived the sinking of the Titanic.

My source code is in my github, which is easier to read

# **Data analysis**

* 1. Firstly, we should import the necessary libraries.

library(ggplot2)

library(gridExtra)

library(grid)

library(randomForest)

library(party)

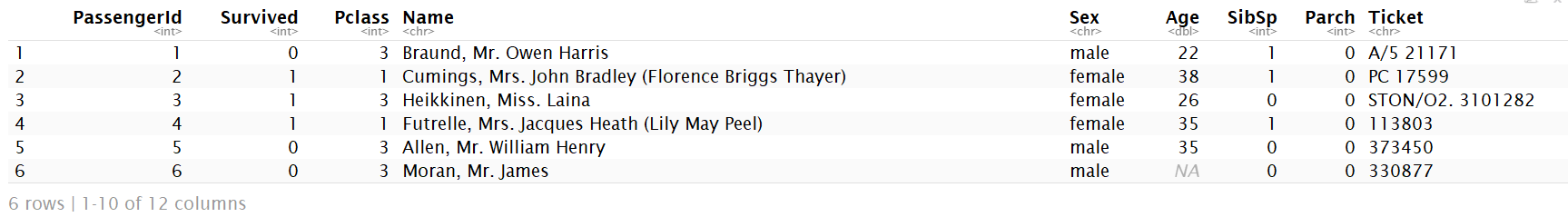
library(rpart)

* 1. Let’s read the training data.

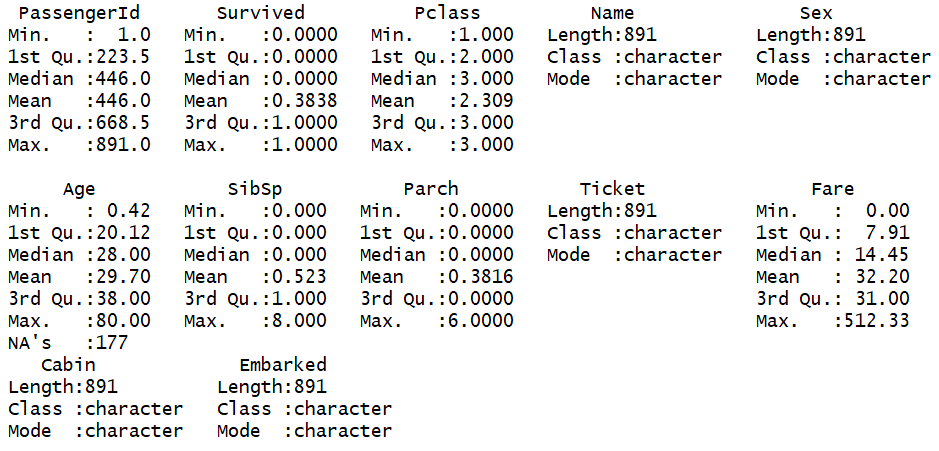
trainingData <- read.csv(file = 'train.csv',stringsAsFactors = FALSE,header = T)

dim(trainingData)

[1] 891 12

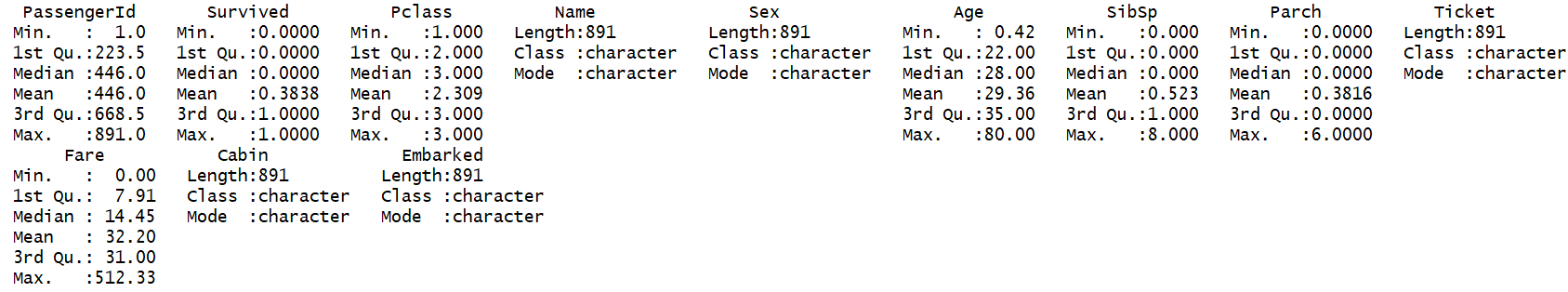
* 1. Let’s see what the data looks like.  
     head(trainingData)
  2. See some summary information about the training data, it shows that 177 values are missing in the Age column.

summary(trainingData)



* 1. So as you see in the summary, there are 177 missing values in Age column, before we visualize the data, we can just simply fill these missing ages with the median age of the dataset.

trainingData[is.na(trainingData$Age),'Age'] <- median(trainingData$Age,na.rm = TRUE)

summary(trainingData)

Now, the missing ages are gone.

* 1. visualize survival base on the gender

df <- aggregate(data.frame(trainingData$Sex,trainingData$Survived), by = list(trainingData$Sex, trainingData$Survived), FUN = length)

df <- data.frame(df[,1],df[,2],df[,3])

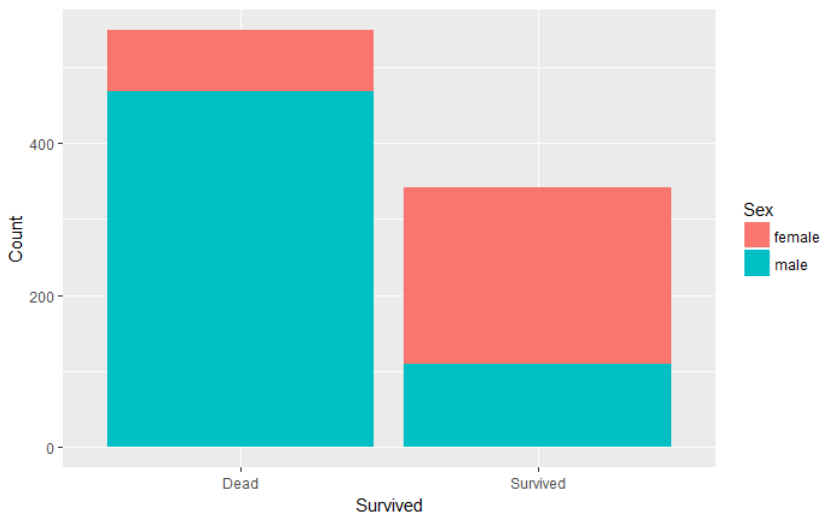
names(df) <- c("Sex","Survived","Count")

df[df[,"Survived"]==1,"Survived"] <- "Survived"

df[df[,"Survived"]==0,"Survived"] <- "Dead"

p <- ggplot(data=df,aes(x=df$Survived,y=df$Count,fill=df$Sex)) + geom\_bar(stat = "identity")

p + labs(x = "Survived",y="Count", fill="Sex")



* 1. Let's now correlate the survival with the age variable

survived <- trainingData$Survived

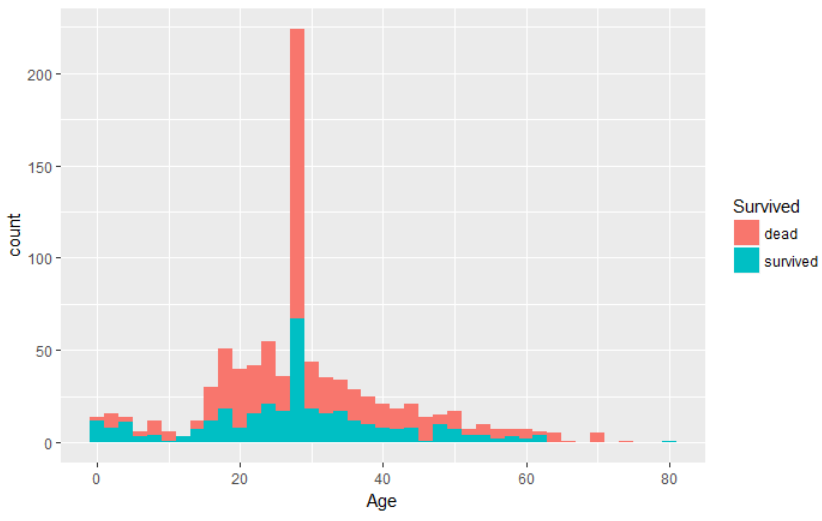
survived[survived==0] <- 'dead'

survived[survived==1] <- 'survived'

df <- data.frame(survived,trainingData$Age)

names(df) <- c("Survived","Age")

ggplot(df, aes(x=Age,fill=Survived)) + geom\_histogram(binwidth = 2)



From the chart above, we can see that passengers who are less than 10 are more likely to survive than older ones who are more than 12 and less than 50. Older passengers seem to be rescued too.

* 1. Next, let's focus on the Fare ticket of each passenger and correlate it with the survival

survived <- trainingData$Survived

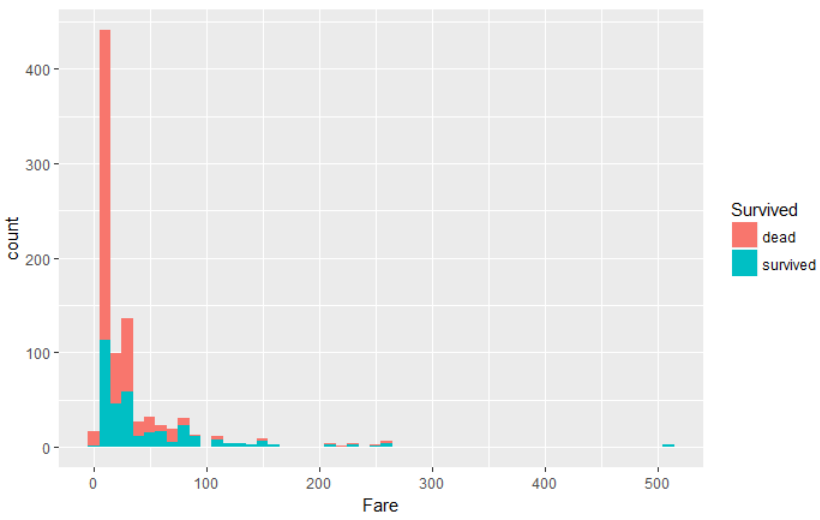
survived[survived==0] <- 'dead'

survived[survived==1] <- 'survived'

df <- data.frame(survived,trainingData$Fare)

names(df) <- c("Survived","Fare")

ggplot(df, aes(x=Fare,fill=Survived)) + geom\_histogram(binwidth = 10)



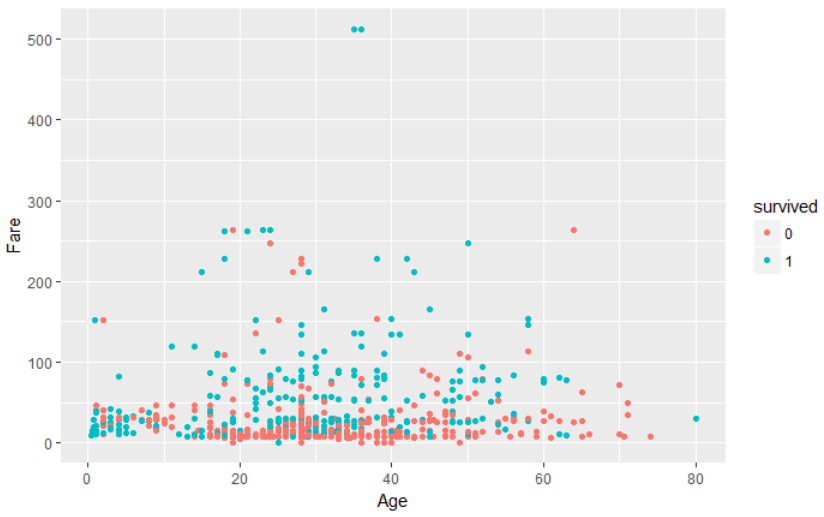
From the chart above, we can see that passengers with cheaper ticket fares are more likely to die. Because passengers with more expensive tickets, and therefore a more import social status, seem to be rescued first.

* 1. Then, combine the age and fare column to visualize the survival on a single chart.

p <- ggplot(trainingData,aes(trainingData$Age,trainingData$Fare)) + geom\_point(aes(colour = survived))

survived <- factor(trainingData$Survived)

p + labs(x = "Age",y="Fare")



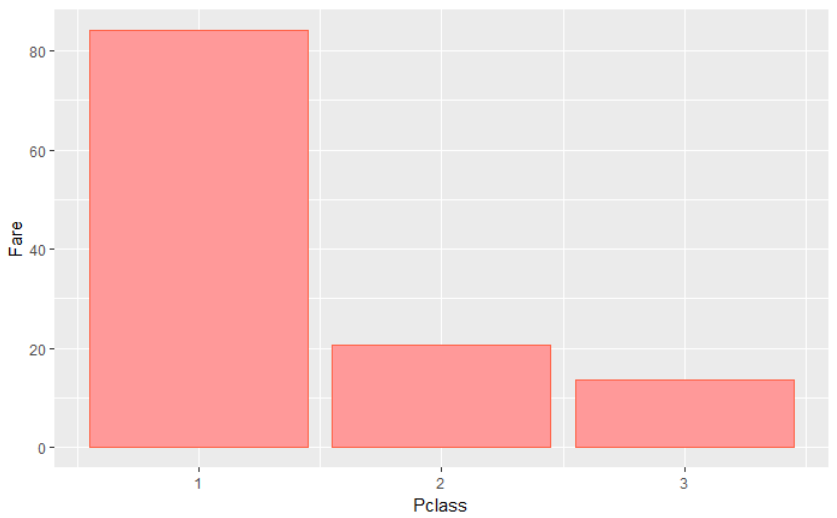
From the scatter plot above, we can see a distinct cluster of dead passengers (the red one) appears on the chart. Those people are adults (age between 15 and 50) of lower class (lowest ticket fares).

* 1. Let’s double check if better ticket class has higher fare.

agg <- aggregate(trainingData$Fare~trainingData$Pclass, trainingData,mean)

names(agg) <- c("Pclass","Fare")

ggplot(agg, aes(agg$Pclass, agg$Fare)) + geom\_col(colour="tomato1",fill="#FF9999") + labs(x = "Pclass",y="Fare")



Yes, it is, the first class tickets have the higher fare.

* 1. Let's now see how the embarkation site affects the survival.

trainingData[trainingData$Embarked=='',"Embarked"] <- 'S'

df <- aggregate(data.frame(trainingData$Embarked,trainingData$Survived), by = list(trainingData$Embarked, trainingData$Survived), FUN = length)

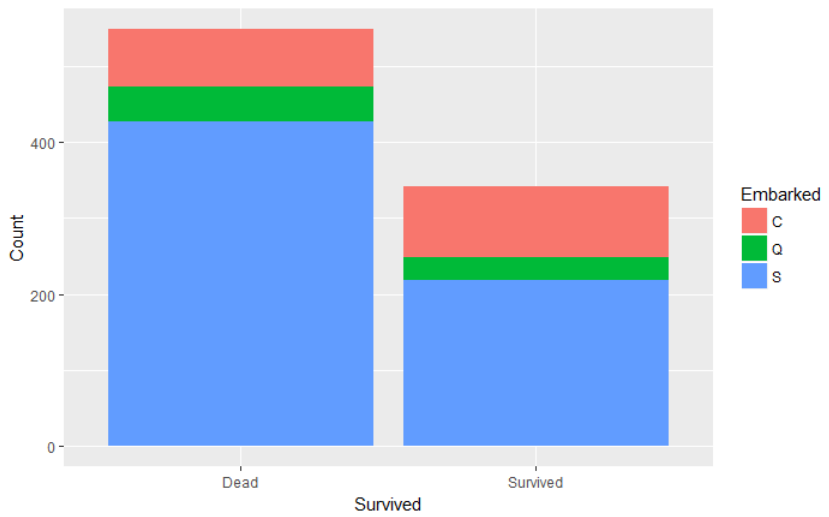
df <- data.frame(df[,1],df[,2],df[,3])

names(df) <- c("Embarked","Survived","Count")

df[df[,"Survived"]==1,"Survived"] <- "Survived"

df[df[,"Survived"]==0,"Survived"] <- "Dead"

ggplot(data=df,aes(x=Survived,y=Count,fill=Embarked)) + geom\_bar(stat = "identity")



There seems to be no distinct correlation here

# Feature engineering

1. In the previous part, we load the data and analyzed the data by creating some ggplot charts. Now we know the most information about the dataset. However, we couldn't manage to analyze more complicated features like the names or the tickets because these required further processing. In this part, I will preprocess the data and to create some new features which can be very useful feeding into the machine learning algorithms.

Let’s combine the training data and testing data together.

getCombinedData <- function(){

trainingData <- read.csv(file = 'train.csv',stringsAsFactors = FALSE,header = T)

testingData <- read.csv(file = 'test.csv',stringsAsFactors = FALSE,header = T)

testingData$Survived <- NA

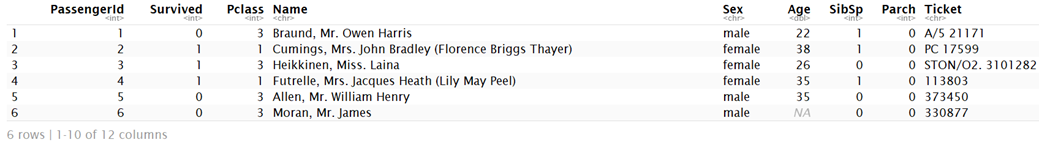
combined <- rbind(trainingData,testingData)

return(combined)

}

combined <- getCombinedData()

head(combined)



1. Extracting the passenger title

When we check the passenger name, we can easily find that the names are consist of first name, title, last name. Something like the following.

* Moran, **Mr**. James
* Allen, **Mr**. William Henry
* Futrelle, **Mrs**. Jacques Heath (Lily May Peel)
* Heikkinen, **Miss**. Laina
* Peter, **Master**. Michael J
* Oliva y Ocana, **Dona**. Fermina

The title is a very important feature of the person; however, the first name and the last name are useless.

getTitles <- function(){

# ",Capt\\.", "\\." means escape sequence. It can be converted to ",Capt.",detect which name contains this string and convert it to "Officer"

titleDictionary = c("Officer","Officer","Officer","Royalty","Royalty","Royalty","Officer","Officer","Royalty",

"Royalty","Mrs","Miss","Mrs","Mr","Mrs","Miss","Master","Royalty")

names(titleDictionary) = c(", Capt\\.",", Col\\.",", Major\\.",", Jonkheer\\.",", Don\\.",", Sir\\.",", Dr\\.",", Rev\\.",

", the Countess\\.",", Dona\\.",", Mme\\.",", Mlle\\.",", Ms\\.",", Mr\\.",", Mrs\\.",", Miss\\.",

", Master\\.",", Lady\\.")

return(titleDictionary)

}

titles <- getTitles()

# convert Name column to Title

convertNameToTitle <- function(titles){

allNames = combined$Name

for (i in 1:length(names(titles))){

index = grep(names(titles)[i],allNames)

allNames[index] = titles[i]

}

return(allNames)

}

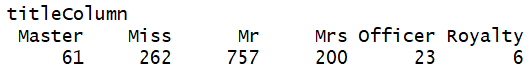
titleColumn <- convertNameToTitle(titles)

# combined$Title <- titleColumn

combined <- cbind(combined[,1:4],Title = titleColumn,combined[,5:ncol(combined)])

# use the function above, all female title will be converted to Mr too.

table(titleColumn)



1. Let’s create family size feature

table(combined$Parch)

getFamilySize <- function(row){

c(row[1] + row[2])

}

familySize <- apply(combined[,c("Parch","SibSp")],1,getFamilySize)

combined$FamilySize <- familySize



1. Analyze the age column

In the first part, I use summary to display the basic information of the whole dataset, from there we can see that there are 177 missing values of Age. That’s a significant portion of our data, we should fill these missing values. However, simply replacing them with the mean or the median age might not be the best solution since the age may differ by groups and categories of passengers.

let's group our dataset by sex, Title and passenger class and for each subset compute the median age.

df <- data.frame(combined$Sex,combined$Pclass,combined$Title,combined$PassengerId,combined$Age,combined$SibSp,combined$Parch,combined$Fare)

names(df) <- c("Sex","Pclass","Title","PassengerId","Age","SibSp","Parch","Fare")

df$Sex <- as.character(df$Sex)

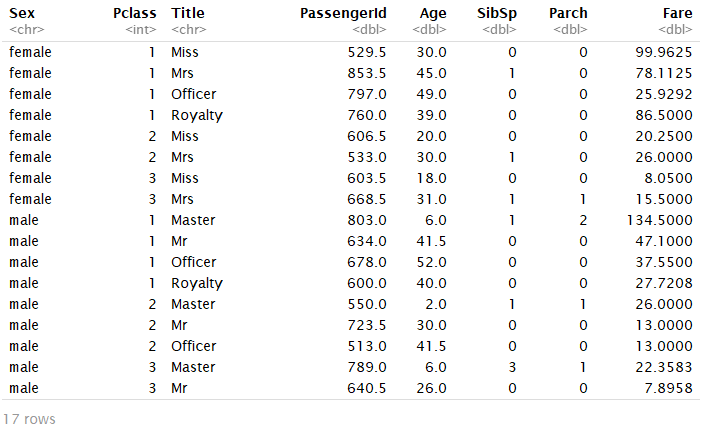
df$Title <- as.character(df$Title)

df <- aggregate(data.frame(df$PassengerId,df$Age,df$SibSp,df$Parch,df$Fare),by = list(df$Title,df$Pclass, df$Sex), FUN = median,na.rm=TRUE)

names(df) <- c("Title","Pclass","Sex","PassengerId","Age","SibSp","Parch","Fare")

df <- df[,c("Sex","Pclass","Title","PassengerId","Age","SibSp","Parch","Fare")]

df



From this table, we can see the median age are different from different gender, Pclass and Title. Then we can fill the missing age by determine which group they are in. We can use decision tree to fill the missing age.

1. Use decision tree to fill the missing age

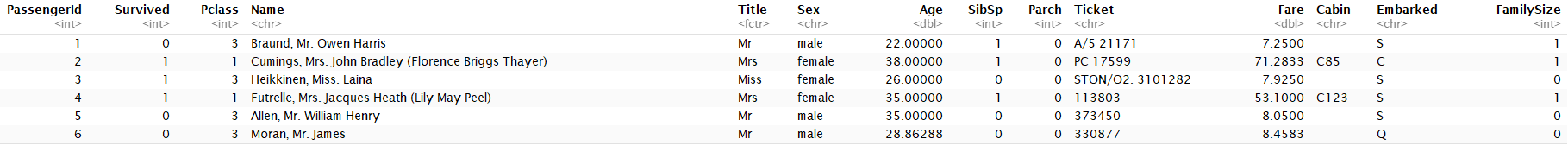
The decision tree works like this: first, find the Sex, Pclass and Title of the person who is missing age. They make decision which age should fill in. For example, from the table above, if this person is female, if her Pclass is 1 and if she is an officer then fill the missing age with 49.

Agefit <- rpart(Age ~ Pclass + Sex + SibSp + Parch + Fare + Embarked + Title,

data=combined[!is.na(combined$Age),],

method="anova")

combined$Age[is.na(combined$Age)] <- predict(Agefit, combined[is.na(combined$Age),])

head(combined)

As you can see, there are no missing values in Age column.

1. There are still one missing Fare in the dataset, let’s fill the missing Fare

boxplot(combined$Fare,ylab = 'Fares',main='Distribution of Fares',col='red')

upper.whisker <- boxplot.stats(combined$Fare)$stats[5]

outlier.filter <- combined$Fare < upper.whisker

fare.equation = "Fare ~ Pclass + Sex + Age + SibSp + Parch + Embarked"

fare.model <- lm(

formula = fare.equation,

data = combined[outlier.filter,]

)

missingFare.row <- combined[

is.na(combined$Fare),

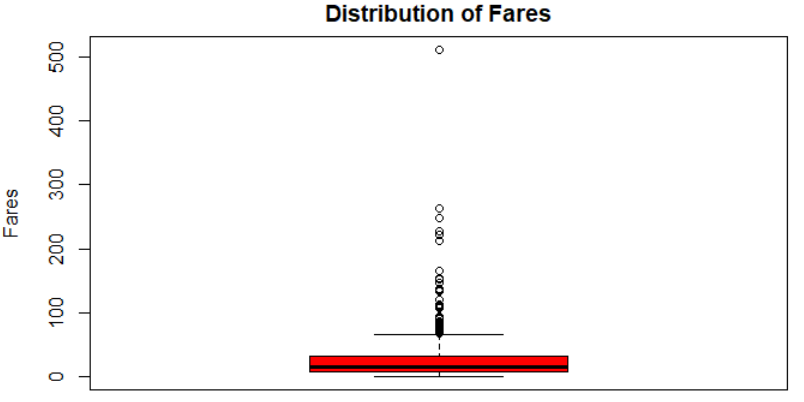
c("Pclass","Sex","Age","SibSp","Parch","Embarked")

]

fare.predictions <- predict(fare.model,missingFare.row)

fare.predictions

combined[is.na(combined$Fare),"Fare"] <- fare.predictions

combined$Fare[1044]

from the boxplot of the fares, we see many outliers, we can get rid these outliers and use regression to predict the missing fare. Then the result is 7.74331.

1. There are 2 missing embarked in the dataset

table(combined$Embarked)



As you can see, there are 2 values do not belong to any category. Because embarked is not an important feature, so let’s just fill them with ‘S’

combined[combined[,"Embarked"]=="","Embarked"] <- 'S'



Then the missing values are gone.

1. Finally, let’s construct a prediction model by using random forest, and use this model to predict the survived column of testing data.

# categorical casting

combined$Sex <- as.factor(combined$Sex)

combined$Cabin <- as.factor(combined$Cabin)

combined$Embarked <- as.factor(combined$Embarked)

# read back training data and testing data from full data

titanic.train <- combined[!is.na(combined$Survived),]

titanic.test <- combined[is.na(combined$Survived),]

titanic.train$Survived <- as.factor(titanic.train$Survived)

survived.equation <- "Survived ~ Age + Fare + Sex + Title + Pclass + SibSp + Parch + Embarked + FamilySize"

survived.formula <- as.formula(survived.equation)

# construct prediction model

titanic.model <- randomForest(formula = survived.formula,data=titanic.train, ntree = 2500, mtry = 3, importance = TRUE)

# check which feature is important

varImpPlot(titanic.model)

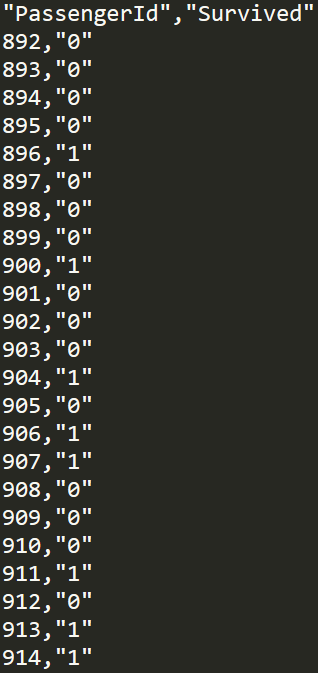
Survived <- predict(titanic.model,newdata = titanic.test)

PassengerId <- titanic.test$PassengerId

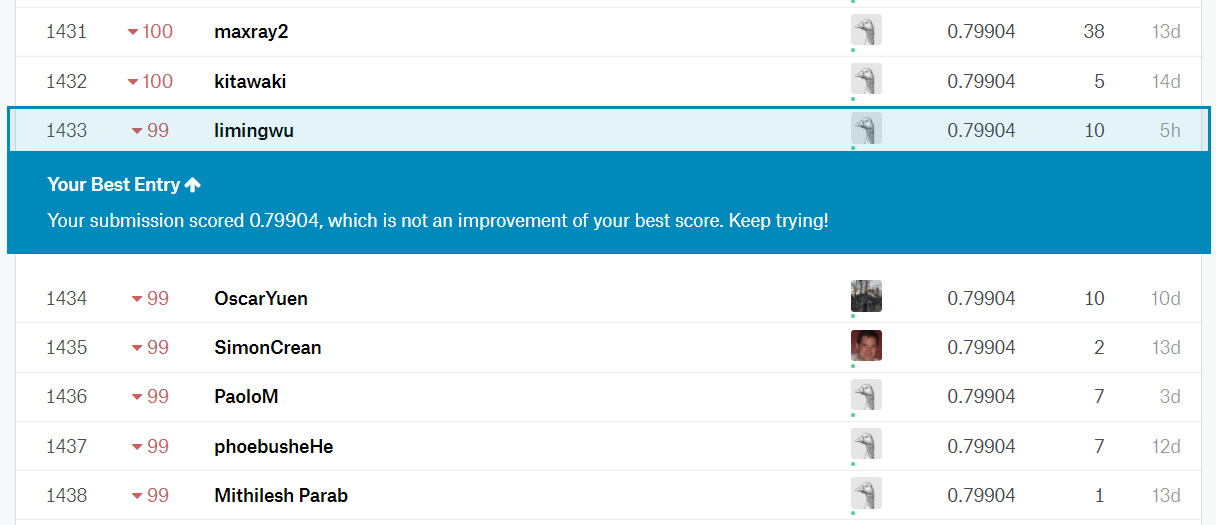
output.df <- data.frame(PassengerId,Survived)

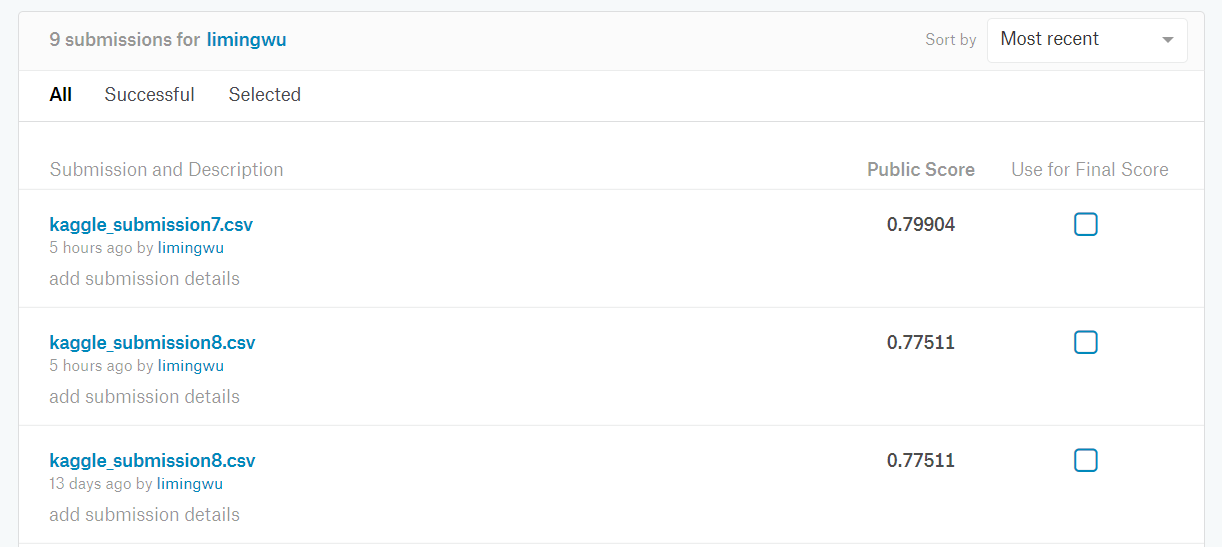
1. Then write the results to a csv file. Our file is something like the following:

write.csv(output.df, file = "kaggle\_submission8.csv", row.names = FALSE)



1. Submit the result file to Kaggle to check my score!



As you can see here, my accuracy is 0.79904, which ranked 1433. Actually I submitted several times which use different algorithms to predict the result, but my highest accuracy is this.