STA 5224: Final Project - Titanic Dataset

Huong Tran

3/27/2022

I. Project Proposal:

Objective:

This project is a competition on Kaggle with target of predicting which passengers survived the Titanic shipwreck by machine learning. Part 1 of the project will focus on cleaning data and obtaining some insights about data, together with variable selection process to create a meaningful dataset that can be used. Model application will be in the second part. Besides the statistical approach in logistic models, other machines learning model will be used. A typical model for supervised learning is decision tree, which is a series of sequential decisions represented as a tree to get a specific result. However, decision tree are prone to overfitting, especially when the tree is deep and random forest is a solution for this kind of problem. In general, a random forests model will creates several random decision trees and aggregate their result. Also, SVM works well with classification and regression, therefore it is good to represent SVM. Finally, the comparsions between these models will be derived. In the last section, statistical inference and interpretation from logistic models will be discussed.

About the dataset:

The data is obtained from the Titanic competition from Kaggle. While the test.csv and gender_submission.csv will be used for model training, the train.csv will be used to evaluate model performance.

The dependence variable is "Survived", which has value 0 or 1, indicates that the person survived after the disaster or not. The others are exploratory variables, with their meaning can be find at the website.

```
test.org <- read.csv2(
  "/Users/huongtran/OU /Course Work/SES 4/STA5224/Final Project 2/data/test.csv",
  header = T, sep = ",")

survive <- read.csv(
  "/Users/huongtran/OU /Course Work/SES 4/STA5224/Final Project 2/data/gender_submission.csv")

train.org <- read.csv2(
  "/Users/huongtran/OU /Course Work/SES 4/STA5224/Final Project 2/data/train.csv",
  header = T, sep = ","
  )

summary(train.org)
```

```
##   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         Length:891         Min.   :0.000   Min.   :0.0000
##  Class :character   Class :character   1st Qu.:0.000   1st Qu.:0.0000
##  Mode  :character   Mode  :character   Median :0.000   Median :0.0000
##                                        Mean   :0.523   Mean   :0.3816
##                                        3rd Qu.:1.000   3rd Qu.:0.0000
##                                        Max.   :8.000   Max.   :6.0000
##     Ticket              Fare              Cabin             Embarked
##  Length:891         Length:891         Length:891         Length:891
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```
colnames(train.org)
```

```
##  [1] "PassengerId" "Survived"    "Pclass"      "Name"        "Sex"
##  [6] "Age"         "SibSp"       "Parch"       "Ticket"      "Fare"
## [11] "Cabin"       "Embarked"
```

```
nrow(train.org)
```

```
## [1] 891
```

There are 891 observations and 10 features in this dataset, since PassengerId is unique, we can not extract any information from this variable and "Survived" is the dependent variable.

II. EDA (Exploratory Data Analysis):

```
library(dplyr)
library(tidyr)
library(stringr)
library(ggplot2)
library(gmodels)
library(vcd)
library(caret)
```

```
anyDuplicated(train.org)
```

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

There is no duplicate rows in this dataset.

```
colSums(is.na(train.org))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0           0
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0           0
```

At first, it seems that there is no missing value in this dataset, but in fact, there are some cells having value of empty string, and containing no information, those are considered as missing value.

```
empty.string <- c()
for (i in colnames(train.org)){
```

```
  empty.string <- c(empty.string, sum(train.org[,i] == ""))
}
names(empty.string) <- colnames(train.org)
empty.string
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         177
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0         687           2
```

```
empty.string.test <- c()
for (i in colnames(test.org)){
  empty.string.test <- c(empty.string.test, sum(test.org[,i] == ""))
}


names(empty.string.test) <- colnames(test.org)
empty.string.test
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##           0           0           0           0          86           0
##       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           1         327           0
```

```
cat("Number of missing value in Cabin in train file: " , empty.string["Cabin"],
    "account for ", round(empty.string["Cabin"] / nrow(train.org) * 100, 2) ,
    "% in total observation", "\n ")
```

```
## Number of missing value in Cabin in train file:  687 account for  77.1 % in total observation
##
```

```
cat("Number of missing value in Cabin in test file: " , empty.string.test["Cabin"],
    "account for ",
    round( empty.string.test["Cabin"]/nrow(test.org) * 100, 2),
    "% in total observation")
```

```
## Number of missing value in Cabin in test file:  327 account for  78.23 % in total observation
```

Cabin variables has 77% missing value in train set and 78.23% in test set, therefore I will drop this variable.

```
train <- train.org
test <- test.org
```

1. What about name?

In the variable "name", there is also title of the person, which indicates their social class and profession.

```
train["Title"] <- str_split_fixed(train$Name, ", ", n = 2)[,2]
train["Title"] <- str_split_fixed(train$Title, ". ", n = 2)[, 1]
unique(train$Title)
```

```
##  [1] "Mr"       "Mrs"      "Miss"     "Master"   "Don"      "Rev"
##  [7] "Dr"       "Mme"      "Ms"       "Major"    "Lady"     "Sir"
## [13] "Mlle"     "Col"      "Capt"     "th"       "Jonkheer"
```

The titles relating to army and "Rev" are less likely to survive, we can just merge them in one level as "Official", all of observation in this variables are male.

```
train[which(train$Title %in% c("Major", "Capt", "Col", "Don", "Rev")), c("Survived", "Title", "Sex")]
```

```
##     Survived Title  Sex
## 31         0   Don male
## 150        0   Rev male
## 151        0   Rev male
## 250        0   Rev male
## 450        1 Major male
## 537        0 Major male
## 627        0   Rev male
## 648        1   Col male
## 695        0   Col male
## 746        0  Capt male
## 849        0   Rev male
## 887        0   Rev male
```

```
train["Title"] <- gsub("Major", "Officer", train[,"Title"], fixed = T)
train["Title"] <- gsub("Capt", "Officer", train[,"Title"], fixed = T)
train["Title"] <- gsub("Col", "Officer", train[,"Title"], fixed = T)
train["Title"] <- gsub("Rev", "Officer", train[,"Title"], fixed = T)
train["Title"] <- gsub("Don", "Officer", train[,"Title"], fixed = T)
unique(train$Title)
```

```
##  [1] "Mr"       "Mrs"      "Miss"     "Master"   "Officer"  "Dr"
##  [7] "Mme"      "Ms"       "Lady"     "Sir"      "Mlle"     "th"
## [13] "Jonkheer"
```

```
train[which(train$Title %in% c("Mme", "Mlle", "th", "Jonkheer", "Lady", "Sir")),
      c("Title","Pclass", "Sex", "Age", "Name", "SibSp", "Survived")]
```

```
##        Title Pclass    Sex Age
## 370      Mme      1 female  24
## 557     Lady      1 female  48
## 600      Sir      1   male  49
## 642     Mlle      1 female  24
## 711     Mlle      1 female  24
## 760       th      1 female  33
## 823 Jonkheer      1   male  38
##                                                                 Name SibSp
## 370                                      Aubart, Mme. Leontine Pauline     0
## 557 Duff Gordon, Lady. (Lucille Christiana Sutherland) ("Mrs Morgan")     1
## 600                       Duff Gordon, Sir. Cosmo Edmund ("Mr Morgan")     1
## 642                                                Sagesser, Mlle. Emma     0
## 711                    Mayne, Mlle. Berthe Antonine ("Mrs de Villiers")     0
## 760          Rothes, the Countess. of (Lucy Noel Martha Dyer-Edwards)     0
## 823                                      Reuchlin, Jonkheer. John George     0
##     Survived
## 370        1
## 557        1
## 600        1
## 642        1
## 711        1
## 760        1
## 823        0
```

For the title listed above, they all represent for unmarried women, therefore, we should change them into

"Miss. And"Jonkheer" will be changed into "Mr". Also the only one value of "Lady" and "Sir" are spouse, we will change their title into "Mrs" and "Mr".

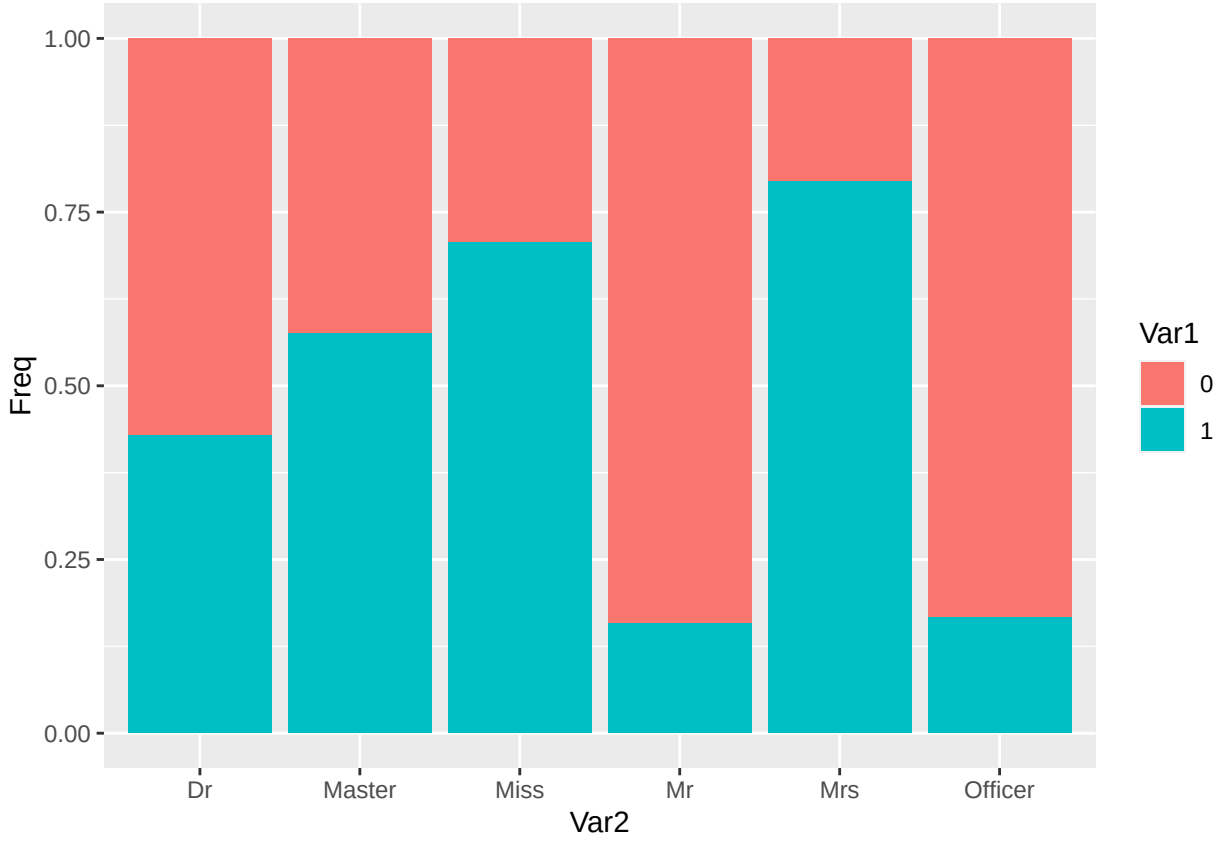
```
train["Title"] <- gsub("Mme", "Miss", train,"Title"], fixed = T)
train["Title"] <- gsub("Mlle", "Miss", train,"Title"], fixed = T)
train["Title"] <- gsub("th", "Miss", train,"Title"], fixed = T)
train["Title"] <- gsub("Jonkheer", "Mr", train,"Title"], fixed = T)
train["Title"] <- gsub("Ms", "Miss", train,"Title"], fixed = T)
train["Title"] <- gsub("Ms", "Miss", train,"Title"], fixed = T)
train["Title"] <- gsub("Ms", "Mrs", train,"Title"], fixed = T)
train["Title"] <- gsub("Lady", "Mrs", train,"Title"], fixed = T)
train["Title"] <- gsub("Sir", "Mr", train,"Title"], fixed = T)

unique(train$Title)
```

```
## [1] "Mr"      "Mrs"     "Miss"    "Master"  "Officer" "Dr"
```

```
df.title <- as.data.frame(table(train$Survived, train$Title))
names(df.title)
```

```
## [1] "Var1" "Var2" "Freq"
```

```
ggplot(df.title, aes(x = Var2, y = Freq, fill = Var1)) + geom_col(position = "fill")
```



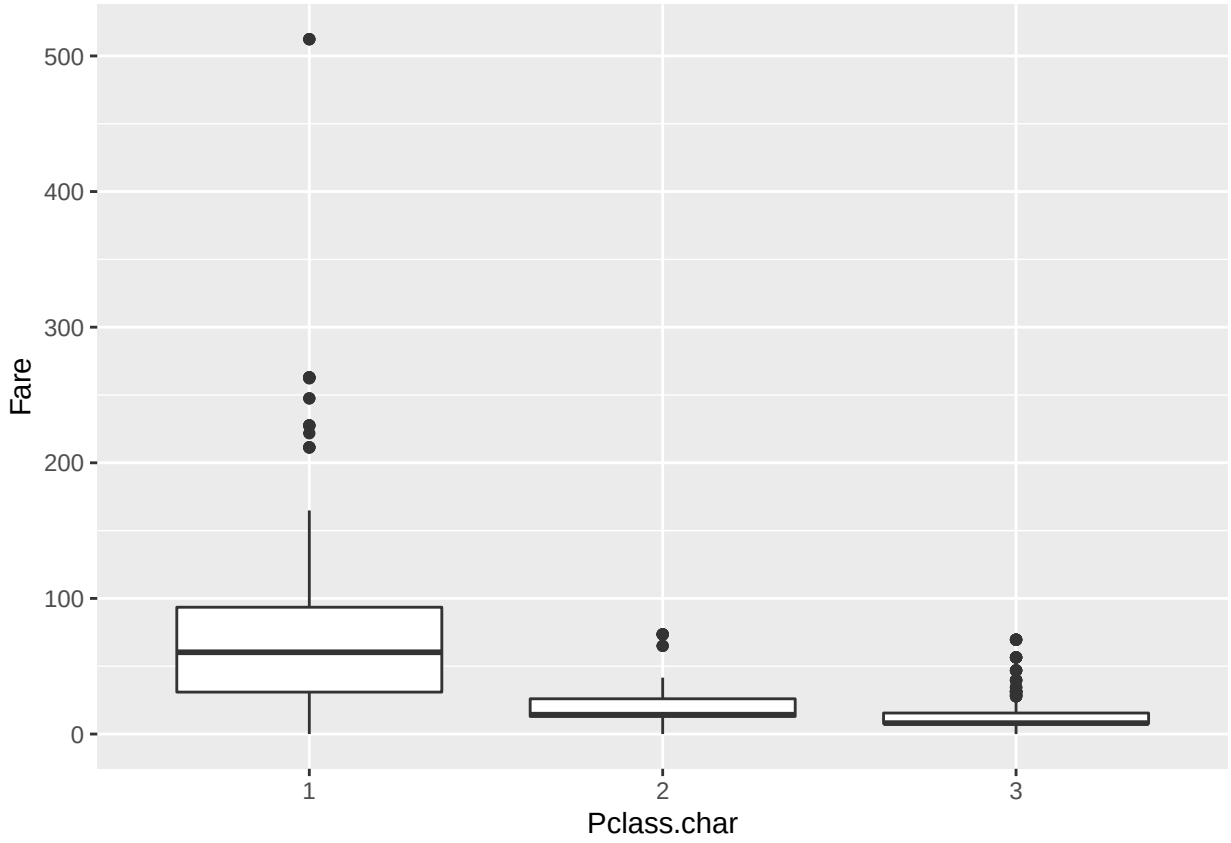
"Miss" and "Mrs" had the highest chance to survived, while "Official" and "Mr" are the two groups with lowest chance to survive.

Also, the information of sex is included in title, there is no need to use the variable sex.

2. What can we Ticket class tell us?

At first, we will look at the relation of Ticket class (*Pclass*) and Passenger fare (*fare*):

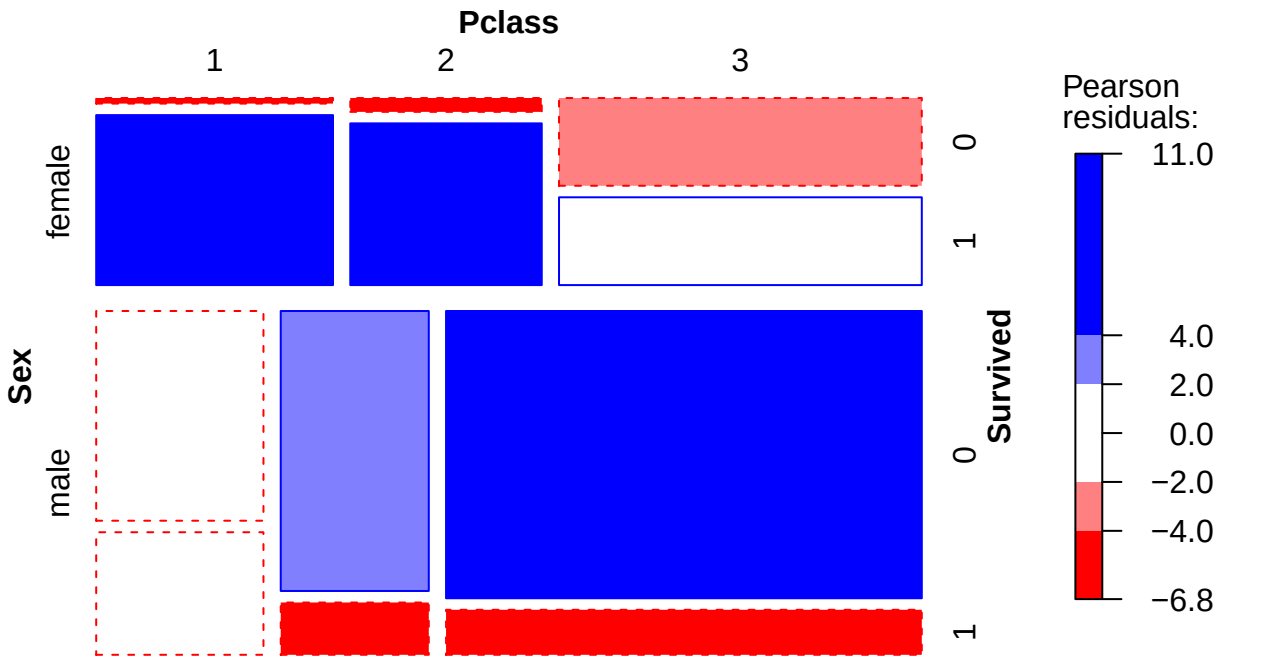
```
train$Fare <- as.numeric(train$Fare)
train["Pclass.char"] <- as.character(train$Pclass)
ggplot(train, aes(Pclass.char, Fare)) + geom_boxplot()
```



Obviously, there is a significant positive correlation of the two variables, the upper class giving the most fare while the lower class giving the least fare.

Keep that in mind, we continue with the difference in their sex:

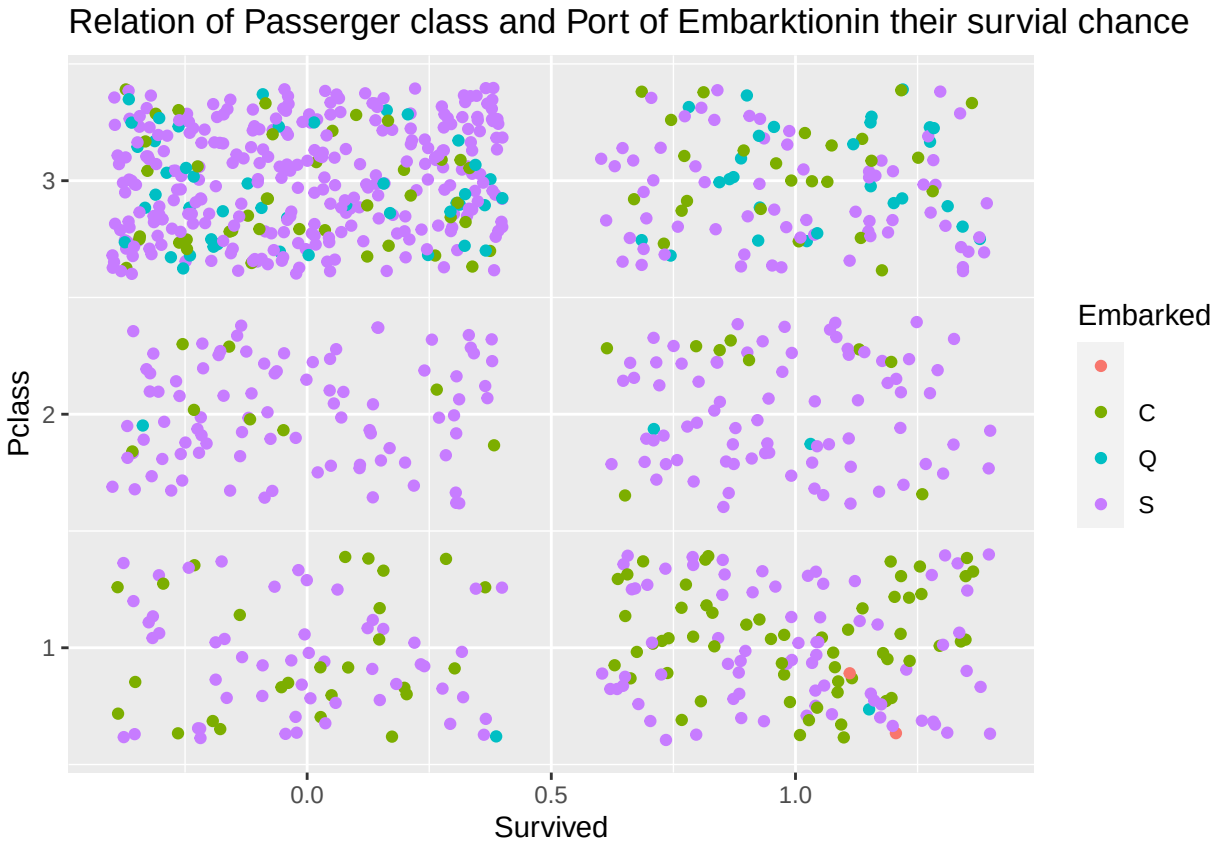
```
mosaic(~ Sex + Pclass + Survived, data = train,  main = "Survival on Titanic",
       gp = shading_Friendly, legend= T )
```



The mosaic plots shows that a women of upper class has the highest chance of survival while a men the the Lower class the lowest chance of survival. Also, although the total number of male is three times the total number of female, but male has the lower probability of survival. In fact, when the disaster hit, women and children were the first priority to go to the rescue boat.

3. Where did they embark?

```
ggplot(train, aes(Survived, Pclass, colour = Embarked)) + geom_jitter() +
  labs(title = "Relation of Passerger class and Port of Embarktionin their survial chance")
```



The majority of passenger used Southampton (S) to embark. However, only lower class (Pclass == 3) used Queenstown (Q).

The two red dots are empty strings, that is why R could not detect any missing value. It turns out their personal information is different, but the others are the same. In fact, this cabin belongs to Mrs. George Nelson, and Miss Amelie is her maid.

```
train[which(train$Embarked == ""),]
```

```
##     PassengerId Survived Pclass                                         Name
## 62           62        1      1                         Icard, Miss. Amelie
## 830         830        1      1 Stone, Mrs. George Nelson (Martha Evelyn)
##        Sex Age SibSp Parch Ticket Fare Cabin Embarked Title Pclass.char
## 62  female  38     0     0 113572   80   B28           Miss           1
## 830 female  62     0     0 113572   80   B28            Mrs           1
```

Since these missing values are from cabin *B28*, let see other variables in deck B:

```
B <- train[grep("B", train$Cabin),]
CrossTable(B$Survived, B$Embarked, prop.c = T, prop.r = F, prop.t = F, prop.chisq = F)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Col Total |
## |-------------------------|
##
##
```

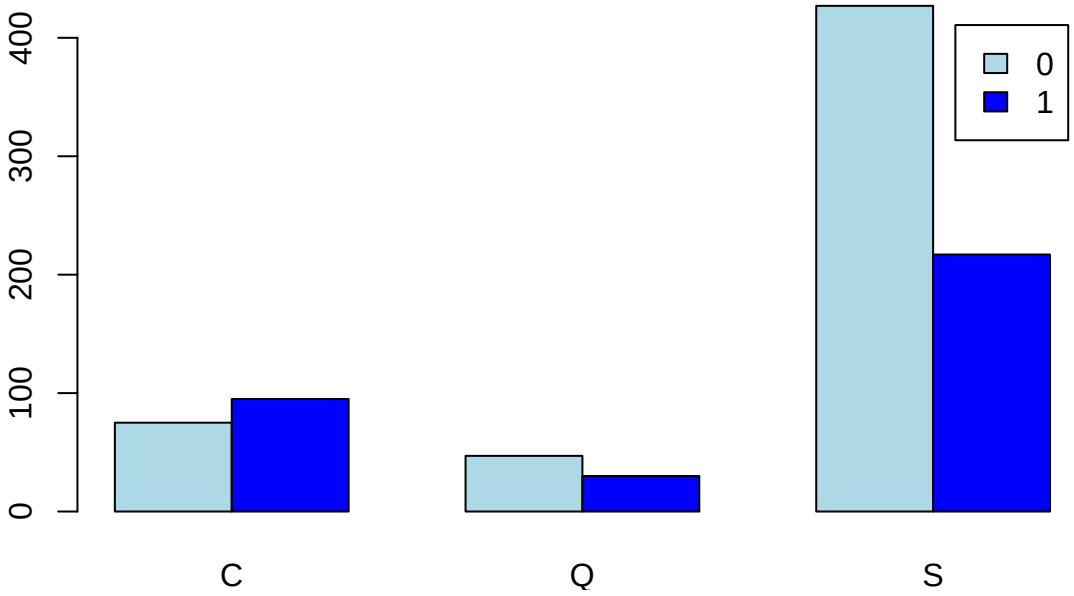
```
## Total Observations in Table:  47
##
##
##              | B$Embarked
##   B$Survived |           |         C |         S | Row Total |
## -------------|-----------|-----------|-----------|-----------|
##            0 |         0 |         5 |         7 |        12 |
##              |     0.000 |     0.227 |     0.304 |           |
## -------------|-----------|-----------|-----------|-----------|
##            1 |         2 |        17 |        16 |        35 |
##              |     1.000 |     0.773 |     0.696 |           |
## -------------|-----------|-----------|-----------|-----------|
## Column Total |         2 |        22 |        23 |        47 |
##              |     0.043 |     0.468 |     0.489 |           |
## -------------|-----------|-----------|-----------|-----------|
##
##
```

In general, passenger with Cabin in deck B used Cherbourg (C) and Southampton (S) as their embarkation. The percentage of survival of port Cherbourg (C) is higher, therefore, we can impute the missing data above by C.

```
train$Embarked[train$Embarked == ""] <- "C"
train[c(62, 830),]
```

```
##     PassengerId Survived Pclass                                      Name
## 62           62        1      1                       Icard, Miss. Amelie
## 830         830        1      1 Stone, Mrs. George Nelson (Martha Evelyn)
##        Sex Age SibSp Parch Ticket Fare Cabin Embarked Title Pclass.char
## 62  female  38     0     0 113572   80   B28        C  Miss           1
## 830 female  62     0     0 113572   80   B28        C   Mrs           1
```

```
table.embarked <- with(train, table(Survived, Embarked))
barplot(table.embarked, beside = T, legend= T, col = c("Lightblue", "blue"))
```



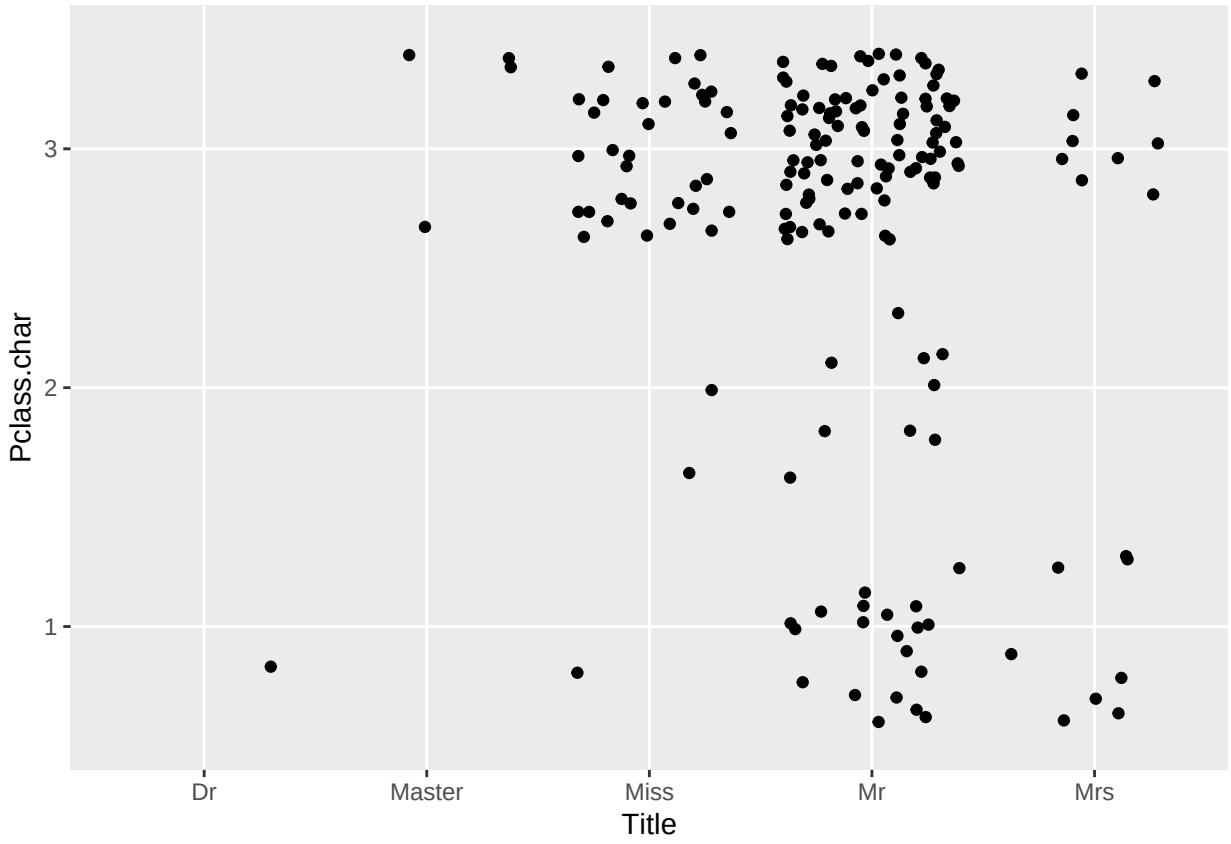
Only at port Cherbourg (C), the percentage of survival is higher.

5. Age:

```
train[,"Age"] <- as.numeric(train[,"Age"])
cat("The number of missing value in variable Age: ",
   empty.string["Age"])
```

```
## The number of missing value in variable Age:  177
```

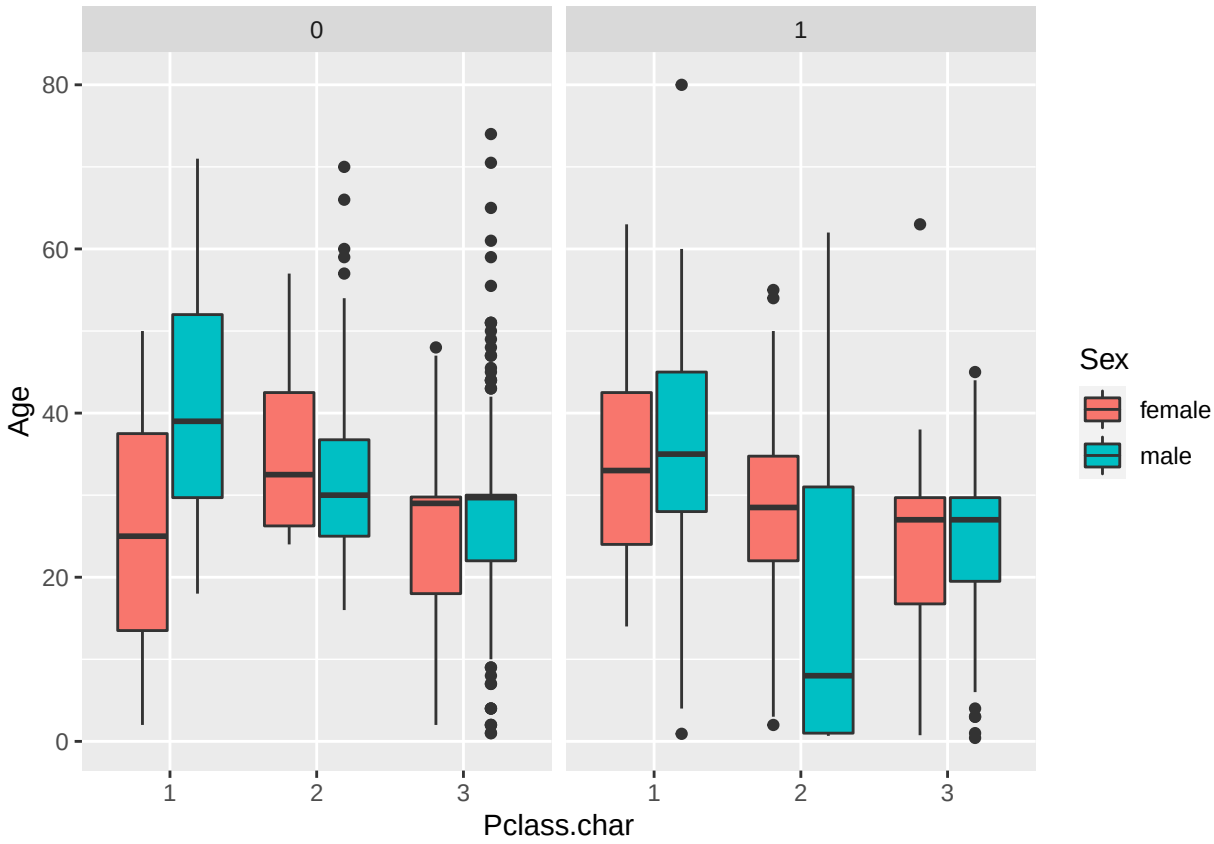
```
age.missing <- train[is.na(train$Age), ]
ggplot(age.missing, aes(Title, Pclass.char)) + geom_jitter()
```



Those missing value are mainly are Mr. and Miss and Mrs, therefore, we can impute the missing data by mean of these group.

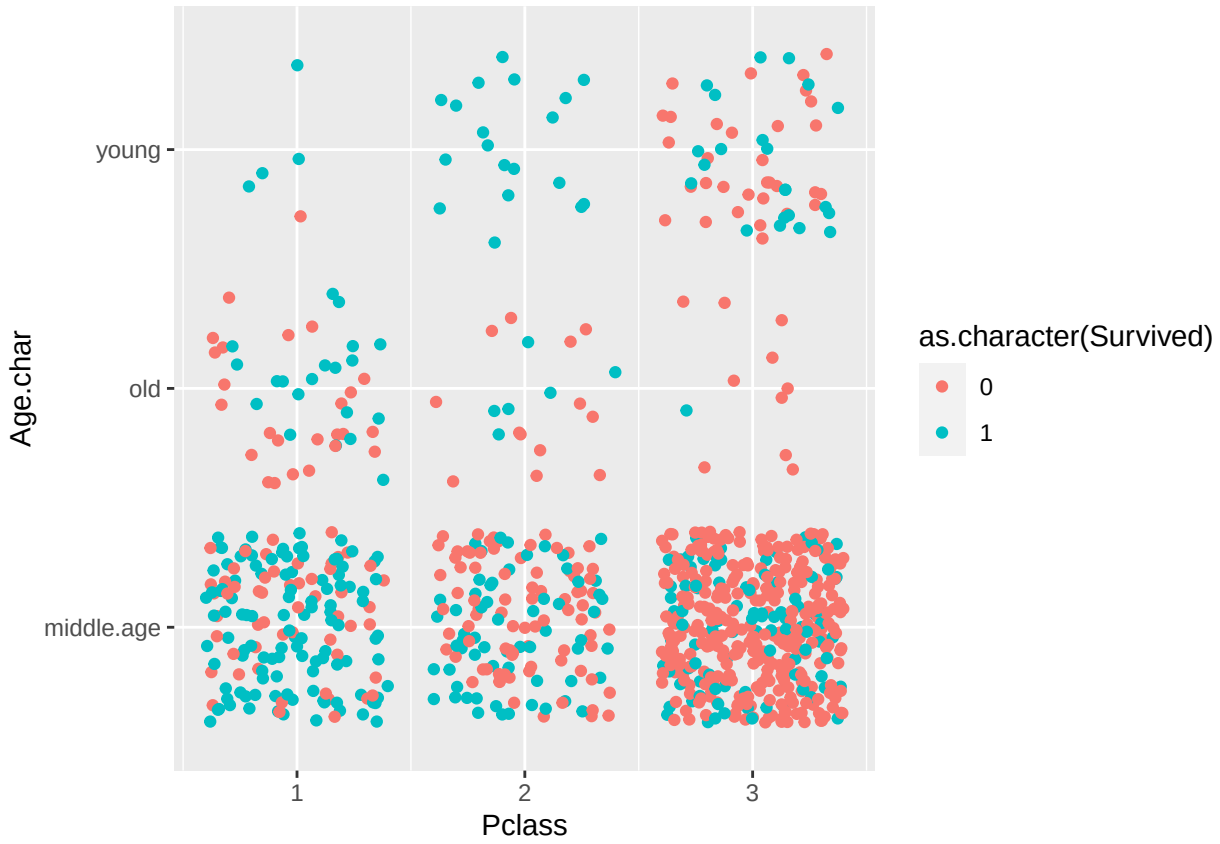
```
m <- mean(train[which(train$Title %in% c("Mr", "Mrs", "Miss")), "Age"])
m <- mean(train[, "Age"], na.rm =  T)
train$Age <- replace_na(train$Age, m)
```

```
#age <- train[which(train$Age != ""),]
ggplot(train, aes(Pclass.char, Age, fill = Sex)) + geom_boxplot() + facet_grid(cols = vars(Survived))
```



Surprisingly, 50% of survival male in middle class was less than 10 years old. Also, more than 50-year-old man is the least likely to survive through the disaster. This suggests a way to divide age into 3 smaller groups:

```
train <- train %>% mutate(Age.char = case_when(Age < 15 ~ "young",
                                               Age < 50 ~ "middle.age",
                                               Age < 100 ~ "old"))
ggplot(train, aes(Pclass, Age.char, colour = as.character(Survived))) + geom_jitter()
```

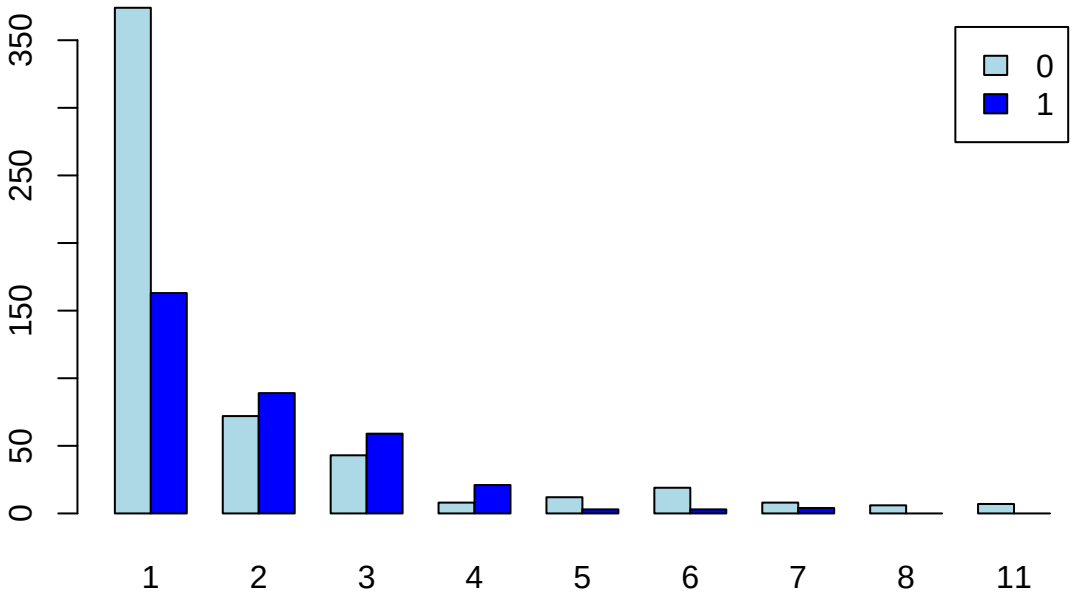


People in the first class has the highest chance to survive, especially when they are in middle age group. In contrast, middle-age men is the most likely died in the disaster. And in fact, the young iin seccond group will survive.

Family size:

At first, both variable *SibSp* and *Parch* contain information about family size, we can create a new variable as *family.size* to obtain information about passenger's family

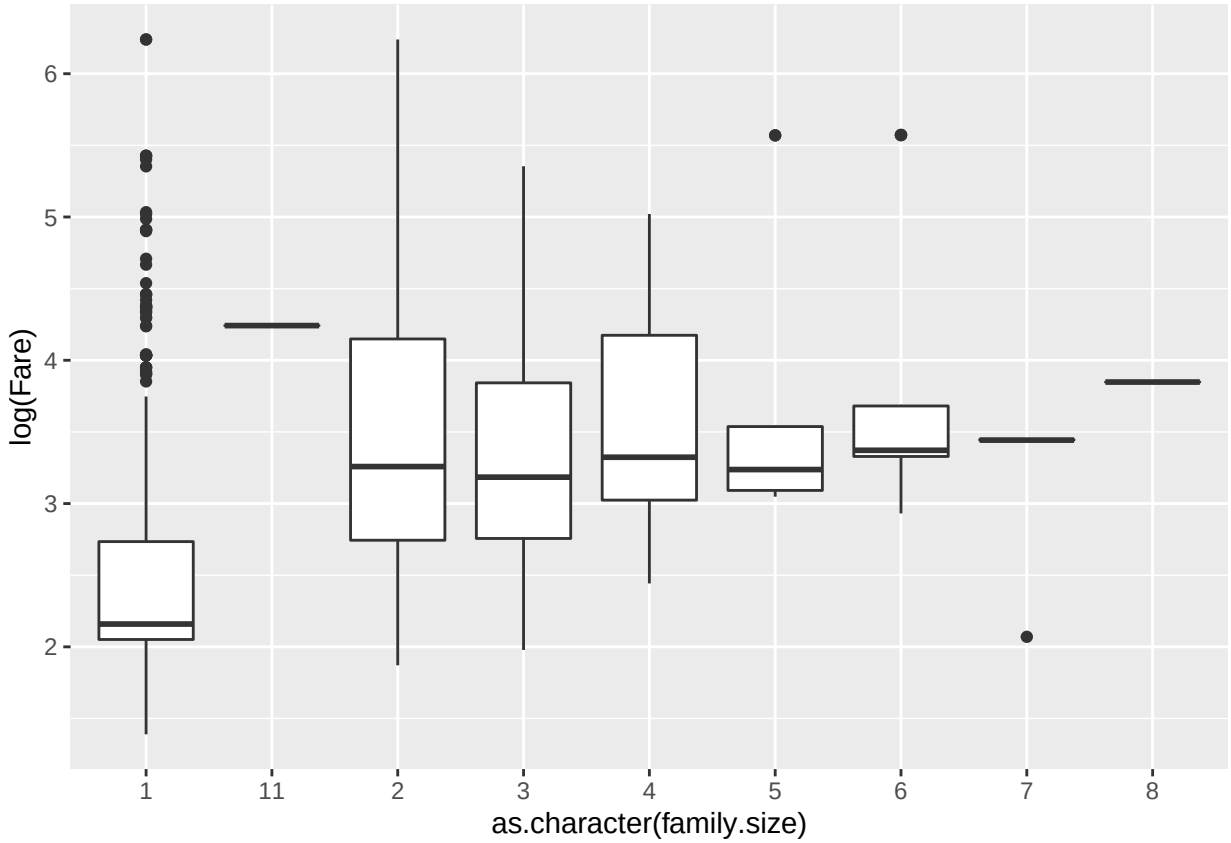
```
train <- train %>% mutate(family.size = SibSp + Parch  +  1 )
table.sib <- with(train, table(Survived, family.size))
barplot(table.sib, beside = T, legend = T, col = c("Lightblue", "Blue"))
```



Surprisingly, the number of survival in family size from 2 to 4 is higher.

```
ggplot(train, aes(as.character(family.size), log(Fare))) + geom_boxplot()
```

```
## Warning: Removed 15 rows containing non-finite values (stat_boxplot).
```



There is positive trend of ln(Fare) and size of family, i.e, the large family size the more fare that ticket they paid. Possibly, it was because the Fare was given the same for all member in a family, not individually different. Remember, when we discuss about missing value of Cabin B28, the two people there had the same fare value. Let's check this logic:

```
fare <- train[which(train$family.size == 2),
              c("Ticket", "Fare", "family.size",
                "Pclass","Name", "Cabin")] %>% arrange(Ticket)
head(fare)
```

```
##   Ticket    Fare family.size Pclass
## 1 110813 75.2500           2      1
## 2 111361 57.9792           2      1
## 3 111361 57.9792           2      1
## 4 113505 55.0000           2      1
## 5 113505 55.0000           2      1
## 6 113509 61.9792           2      1
##                                                 Name Cabin
## 1 Warren, Mrs. Frank Manley (Anna Sophia Atkinson)   D37
## 2                      Hippach, Miss. Jean Gertrude   B18
## 3  Hippach, Mrs. Louis Albert (Ida Sophia Fischer)   B18
## 4           Chibnall, Mrs. (Edith Martha Bowerman)   E33
## 5                       Bowerman, Miss. Elsie Edith   E33
## 6                    Ostby, Mr. Engelhart Cornelius   B30
```

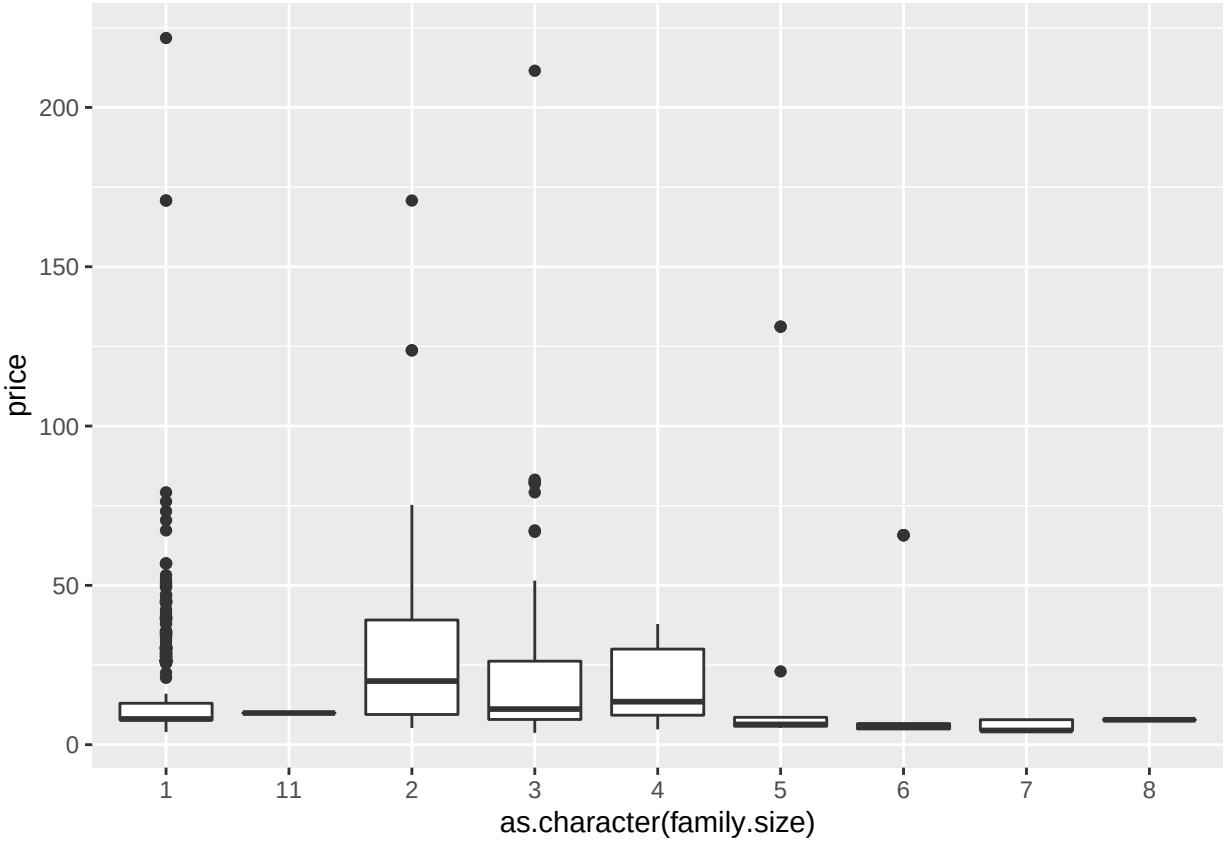
It seems that family members had the same ticket number would have the the same Fare. Now, we will write a function to check how correct this assumption is:

```
train <- train %>% group_by(Ticket) %>% add_count() %>% mutate(mean.fare = mean(Fare))
nrow(train[which(train$Fare != train$mean.fare),])
```

```
## [1] 2
```

As we expected, there are only 2 cases that does not agree with our assumption. Therefore, it is actually a correlation between the family size and fare. To get rid of it, I will find the price that each person has to pay for their ticket and also fill 0 in price by the mean based on their class.

```
train <- train %>% mutate(price = Fare / n )
for (i in 1:3){
  m <- mean(train[which(train$Pclass == i & train$price != 0), "price"]$price)
  train[which(train$Pclass == i & train$price == 0), "price"] <- m
}

ggplot(train, aes(as.character(family.size), price)) + geom_boxplot()
```



Until this point, there is no dependence of ticket fare or price. But the variable "price" is actually dependent on *Pclass*, and this happens in practice when you have to pay more to get the best service.

Variable Selection:

```
##  [1] "PassengerId" "Survived"    "Pclass"      "Name"        "Sex"
##  [6] "Age"         "SibSp"       "Parch"       "Ticket"      "Fare"
## [11] "Cabin"       "Embarked"    "Title"       "Pclass.char" "Age.char"
## [16] "family.size" "n"           "mean.fare"   "price"
```

```
## Adding missing grouping variables: `Ticket`
```

```
##  [1] "Survived"    "Pclass"      "Sex"         "Age"         "SibSp"
##  [6] "Parch"       "Fare"        "Embarked"    "Title"       "Pclass.char"
## [11] "Age.char"    "family.size" "price"
```

```
## Survived ~ Title + Pclass + family.size + Age + Fare + price +
##     Embarked
```

Forward selection suggest the model including *Title, Pclass, family.size, Age, Fare, price, Embarked* as predictors. This model produced the AIC of 752.1, which is the same AIC as the smaller model with *Title, Pclass, family.size, Age, Fare, price*. Therefore, I will drop *Embarked* out of my model.

```
keep <- c("Survived","Title", "Pclass", "family.size", "Age", "Fare", "price")
```

Test Preparation:

The same way of data cleaning for test set:

```
# Title feature:

test["Title"] <- str_split_fixed(test$Name, ", ", n = 2)[,2]
test["Title"] <- str_split_fixed(test$Title, ". ", n = 2)[, 1]
test["Title"] <- gsub("Major", "Officer", test[,"Title"], fixed = T)
test["Title"] <- gsub("Capt", "Officer", test[,"Title"], fixed = T)
test["Title"] <- gsub("Col", "Officer", test[,"Title"], fixed = T)
test["Title"] <- gsub("Rev", "Officer", test[,"Title"], fixed = T)
test["Title"] <- gsub("Don", "Officer", test[,"Title"], fixed = T)
test["Title"] <- gsub("Mme", "Miss", test[,"Title"], fixed = T)
test["Title"] <- gsub("Mlle", "Miss", test[,"Title"], fixed = T)
test["Title"] <- gsub("th", "Miss", test[,"Title"], fixed = T)
test["Title"] <- gsub("Jonkheer", "Mr", test[,"Title"], fixed = T)
test["Title"] <- gsub("Ms", "Miss", test[,"Title"], fixed = T)
test["Title"] <- gsub("Ms", "Miss", test[,"Title"], fixed = T)
test["Title"] <- gsub("Ms", "Mrs", test[,"Title"], fixed = T)
test["Title"] <- gsub("Lady", "Mrs", test[,"Title"], fixed = T)
test["Title"] <- gsub("Sir", "Mr", test[,"Title"], fixed = T)
unique(test$Title)
```

```
## [1] "Mr"       "Mrs"      "Miss"     "Master"   "Officer"  "Dr"       "Officera"
```

```
#This title is only in test set:
test["Title"] <- gsub("Officera", "Officer", test[,"Title"], fixed = T)
unique(test$Title)
```

```
## [1] "Mr"      "Mrs"     "Miss"    "Master"  "Officer" "Dr"
```

```
test$Fare <- as.numeric(test$Fare)
test$Age <- as.numeric(test$Age)

#age.missing.test <- test[is.na(test$Age), ]
#ggplot(age.missing.test, aes(Pclass, Sex)) + geom_jitter()
#m.test <- mean(test[which(test$Pclass == 3), "Age"], na.rm =  T)
test$Age <- replace_na(test$Age, mean(test$Age))

test <- test %>% mutate(family.size = SibSp + Parch  +  1 )
test <- test %>% group_by(Ticket) %>% add_count() %>% mutate(mean.fare = mean(Fare))

test <- test %>% mutate(price = Fare / n )
for (i in 1:3){
  m <- mean(test[which(test$Pclass == i & test$price != 0), "price"]$price)
  test[which(test$Pclass == i & test$price == 0), "price"] <- m
}
```

Models:

```
train.mod <- train[, keep]
keep.test <- keep[-1]
test.mod <- test[, keep.test]
```

1. Logistic Model:

```
mod.reg <- glm(Survived ~., data = train.mod, family = binomial)
mod.fit <- as.numeric(fitted(mod.reg) >  0.4)
fit.tab <- xtabs(~ train.mod$Survived + mod.fit)
reg.acc.fit  <- (fit.tab[1,1] + fit.tab[2,2]) / sum(fit.tab)
summary(mod.reg)
```

```
##
## Call:
## glm(formula = Survived ~ ., family = binomial, data = train.mod)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5124  -0.5815  -0.3972   0.5141   2.5959
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.895850   0.978740   2.959 0.003089 **
## TitleMaster   2.909533   0.951460   3.058 0.002228 **
## TitleMiss     2.404564   0.835501   2.878 0.004002 **
## TitleMr      -0.647279   0.806985  -0.802 0.422498
## TitleMrs      3.142493   0.848613   3.703 0.000213 ***
## TitleOfficer -1.046988   1.122342  -0.933 0.350892
## Pclass       -1.142484   0.162948  -7.011 2.36e-12 ***
## family.size  -0.501399   0.085670  -5.853 4.84e-09 ***
## Age          -0.023712   0.008951  -2.649 0.008071 **
## Fare          0.008480   0.003949   2.147 0.031780 *
## price        -0.012403   0.009034  -1.373 0.169741
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  730.13  on 880  degrees of freedom
## AIC: 752.13
##
## Number of Fisher Scoring iterations: 5
```

2. Decision Tree:

- Decision Tree is a supervised learning method.
- Decision Tree is a graph to represent choices and their results in a form of a tree.

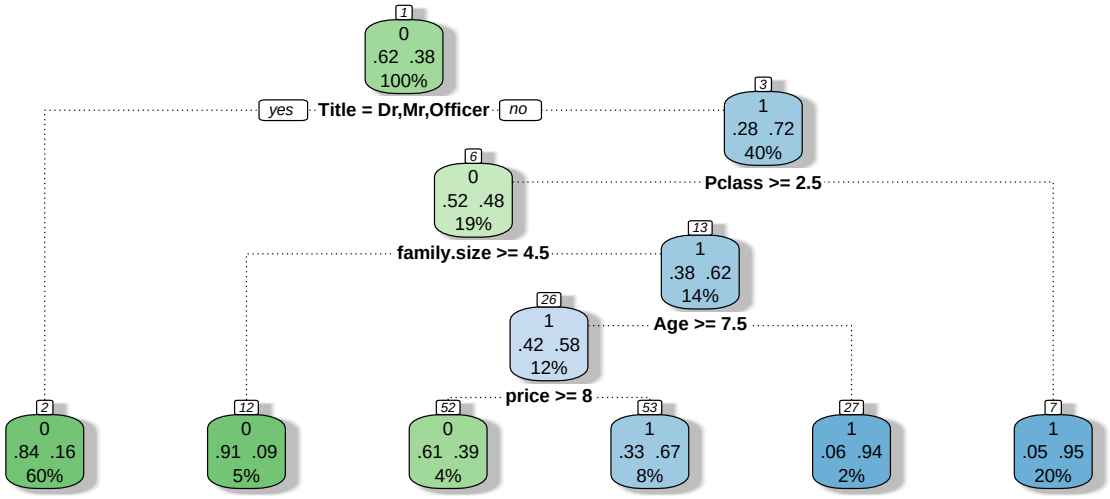
```
library(rpart)
library(rattle)
```

```
## Loading required package: tibble
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(rpart.plot)
tree <- rpart(Survived ~., data = train.mod, method = "class")
fancyRpartPlot(tree)
```



Rattle 2022−Apr−01 15:33:58 huongtran

```
tree.fit <- predict(tree, train.mod, type = "class")
tree.tab.fit <- table(train.mod$Survived, tree.fit)
tree.acc.fit <- sum(diag(tree.tab.fit)) / sum(tree.tab.fit)
```

3. Random Forest:

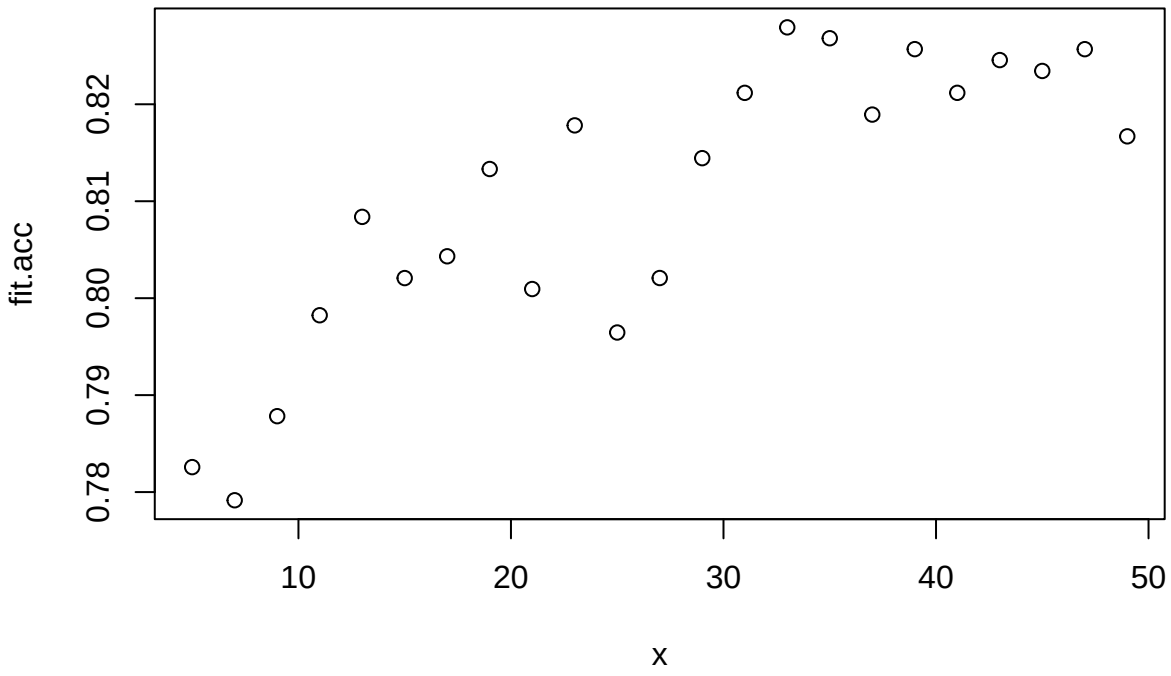
- Random forest creates several random Decision Tree output is the aggregate of those trees.

```
library(randomForest)
```

```
## randomForest 4.7-1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:rattle':
##
##     importance
## The following object is masked from 'package:ggplot2':
##
##     margin
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
set.seed(456)
fit.acc <- c()
x  <- c()
```

```r
for (i in  seq(5, 50, by = 2)){
  rf <- randomForest(as.factor(Survived) ~., data = train.mod, ntree = i)
  conf.matrix <- rf$confusion
  fit.acc <- c(fit.acc,  sum(diag(conf.matrix)) / sum(conf.matrix))
  x <- c(x, i)
}

plot(x, fit.acc)
```




```r
num.tree <- x[which.max(fit.acc)]
```

```r
library(randomForest)
rf <- randomForest(as.factor(Survived) ~., data = train.mod, ntree = num.tree)
rf.acc.fit <- sum(diag(rf$confusion)) / sum(rf$confusion)
```

SVM:

- SVM (Support Vector Machine) is a supervised learning methods, which used to classified data.
- Figure in wiki

```r
library(e1071)

svm.mod <- svm(as.factor(Survived) ~., data = train.mod, scale =  F)
svm.tab <- table(as.character(train.mod$Survived), svm.mod$fitted)
svm.fit.acc <- sum(diag(svm.tab)) / sum(svm.tab)

#make prediction:
```

Comparsions:

```r
library(xtable)
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
reg.pred <-  as.numeric(predict(mod.reg, test.mod) > 0.4)
reg.tab.pred <- xtabs(~ survive$Survived + reg.pred)
reg.acc.pred  <- (reg.tab.pred [1,1] + reg.tab.pred [2,2]) / sum(reg.tab.pred)


tree.pred <- predict(tree, test.mod, type = "class")
tree.tab.pred <- xtabs(~ survive$Survived + tree.pred)
tree.acc.pred  <- (tree.tab.pred [1,1] + tree.tab.pred [2,2]) / sum(tree.tab.pred)

rf.pred <- predict(rf, test.mod)
rf.tab.pred <- xtabs(~survive$Survived + rf.pred)
rf.acc.pred <- (rf.tab.pred [1,1] + rf.tab.pred [2,2]) / sum(rf.tab.pred)

#sum.pred <- predict(sum.mod, test.mod)
r <- data.frame(logistic = c(reg.acc.fit, reg.acc.pred),
                tree = c(tree.acc.fit, tree.acc.pred),
                random.forest = c(rf.acc.fit, rf.acc.pred))

rownames(r) <- c("Fit accuarcy", "Prediction accuracy")
kable(r, align = "c", "latex", booktabs=T, escape = F) %>%
  kable_styling(latex_options = "HOLD_position", position = "left") %>%
  row_spec(1, hline_after = T) %>%
  add_header_above(c(" ", "Summary results" = 2, ""), bold =  T, italic = T)
```

	Summary results		
	logistic	tree	random.forest
Fit accuarcy	0.8260382	0.8428732	0.8223099
Prediction accuracy	0.9184290	0.8923445	0.8761329