Titanic Analysis

Taras the Analyst

2023-04-09

Introduction

This is the report produced from the Kaggle notebook 'Titanic Analysis' by Taras K. from 03/18/2023.

The original inspirational source is by Hilla Behar

In this analysis the following questions were asked:

1. What is the relationship the features and a passenger's chance of survival.
2. Prediction of survival for the entire ship.

Last update: 09/04/2023 (see the list of updates at the end of this work)

Setting the environment

Packages

```
# The following packages are to be used for the current analysis
library(dplyr)         # for data manipulation
library(tidyverse)     # for working operations
library(ggplot2)       # for data visualization
library(GGally)        # Extension to 'ggplot2'
library(rpart)         # decision tree model package
library(rpart.plot)    # decision tree visualization package
library(ggcorrplot)    # to understand the correlation matrix
library(randomForest)  # planting the trees needs some methodology...:)
library(pander)        # to create pretty tables
library(knitr)         # to create pretty tables
library(tinytex)       # to use the features for file rendering to .pdf
```

Loading the data sources

```
test <- read.csv('./test.csv',stringsAsFactors = FALSE)
train <- read.csv('./train.csv', stringsAsFactors = FALSE)
dim(test)   # check the test data frame dimensions
```

```
## [1] 418  11
```

```
dim(train)  # check the train data frame dimensions
```

```
## [1] 891  12
```

Data elaboration

Merging both datasets into a consolidated one*

bind_rows() is to be used, as **rbind()** doesn't work here due to different number of columns in *train* and *test*

```
full <- bind_rows(train,test)
dim(full)  # check the resulted data frame dimensions
```

```
## [1] 1309   12
```

```
str(full)  # check the resulted data frame structure
```

```
## 'data.frame':    1309 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)" "Heik
##  $ 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" ...
```

The data is to be checked for missing values

```
## [1] "Here is missing value check:"
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0         418           0           0           0         263
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           1           0           0
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0          NA           0           0           0          NA
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0          NA        1014           2
```

So, the ouput is: N/As - left table, NULLs - right table

```
knitr::kable(list(k1, k2))
```

```
# cross-checking the empty records for Embarked
filter(full, full$Embarked == "")
```

```
##   PassengerId Survived Pclass                                      Name    Sex
## 1          62        1      1                       Icard, Miss. Amelie female
## 2         830        1      1 Stone, Mrs. George Nelson (Martha Evelyn) female
##   Age SibSp Parch Ticket Fare Cabin Embarked
## 1  38     0     0 113572   80   B28
## 2  62     0     0 113572   80   B28
```

	x
PassengerId	0
Survived	418
Pclass	0
Name	0
Sex	0
Age	263
SibSp	0
Parch	0
Ticket	0
Fare	1
Cabin	0
Embarked	0

	x
PassengerId	0
Survived	NA
Pclass	0
Name	0
Sex	0
Age	NA
SibSp	0
Parch	0
Ticket	0
Fare	NA
Cabin	1014
Embarked	2

```r
# getting it into a bit more visually attractive way
kable(filter(full, full$Embarked == ""))
```

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

```r
# getting the digits of missing values
paste("= N/A in full dataset:")  # that's added for some internal explanations
```

```
## [1] "= N/A in full dataset:"
```

```r
pander(table(is.na(full)))  # showing aggregated "n/a" values within each column
```

FALSE	TRUE
15026	682

Cleaning & transforming the data

```r
full$Embarked[full$Embarked == ""] = "C"
```

Change the empty strings in Embarked to the first choice "C"

```r
apply(full, 2, function(x) length(unique(x)))
```

See how many features can be transformed to factors

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##        1309           3           3        1307           2          99
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           7           8         929         282         187           3
```

Move the attributes Survived, Pclass, Sex, Embarked to be factors

```
cols <- as.factor(c("Survived", "Pclass", "Sex", "Embarked"))
for (i in cols){
  full[, i] <- as.factor(full[, i])
}
```

```
str(full)
```

Now let's look on the structure of the full data set

```
## 'data.frame':    1309 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass     : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)" "Heik
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ 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   : Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
```

Move the attributes Survived, Pclass, Sex, Embarked to be factors within train data set

```
cols <- as.factor(c("Survived", "Pclass", "Sex", "Embarked"))
for (i in cols){
  train[, i] <- as.factor(train[, i])
}
```

Now let's look on the structure of the train data set str(train)

Analyse the cleaned data

The data has been loaded & cleaned a little bit so far. Now, it's time to look at the relationships between the different attributes within set and to check the correlations within factored attributes, so to see if there's something useful.

```
full_fctrs <- full[, c("Survived", "Pclass", "Sex", "Embarked")]
train_fctrs <- train[, c("Survived", "Pclass", "Sex", "Embarked")]
```

```
dim(full_fctrs)  # check if the re-shaping went well resulted in 4 columns only
```

```
## [1] 1309    4
```

```
dim(train_fctrs)
```

```
## [1] 891   4
```

```r
str(train_fctrs)  # getting the structure overview of train factors
```

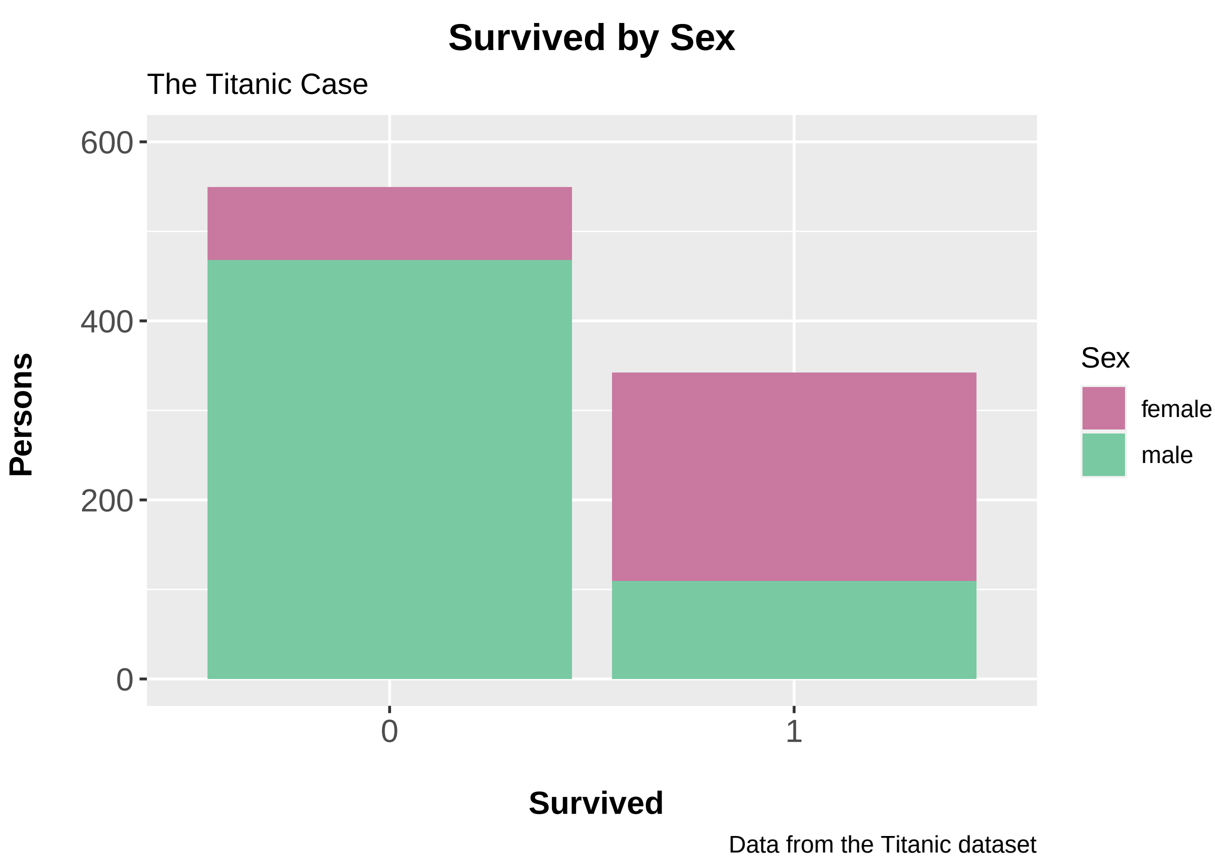
```
## 'data.frame':    891 obs. of  4 variables:
##  $ Survived: Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass  : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...
##  $ Sex     : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ Embarked: Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

```r
str(full_fctrs)   # getting the structure overview of test factors
```

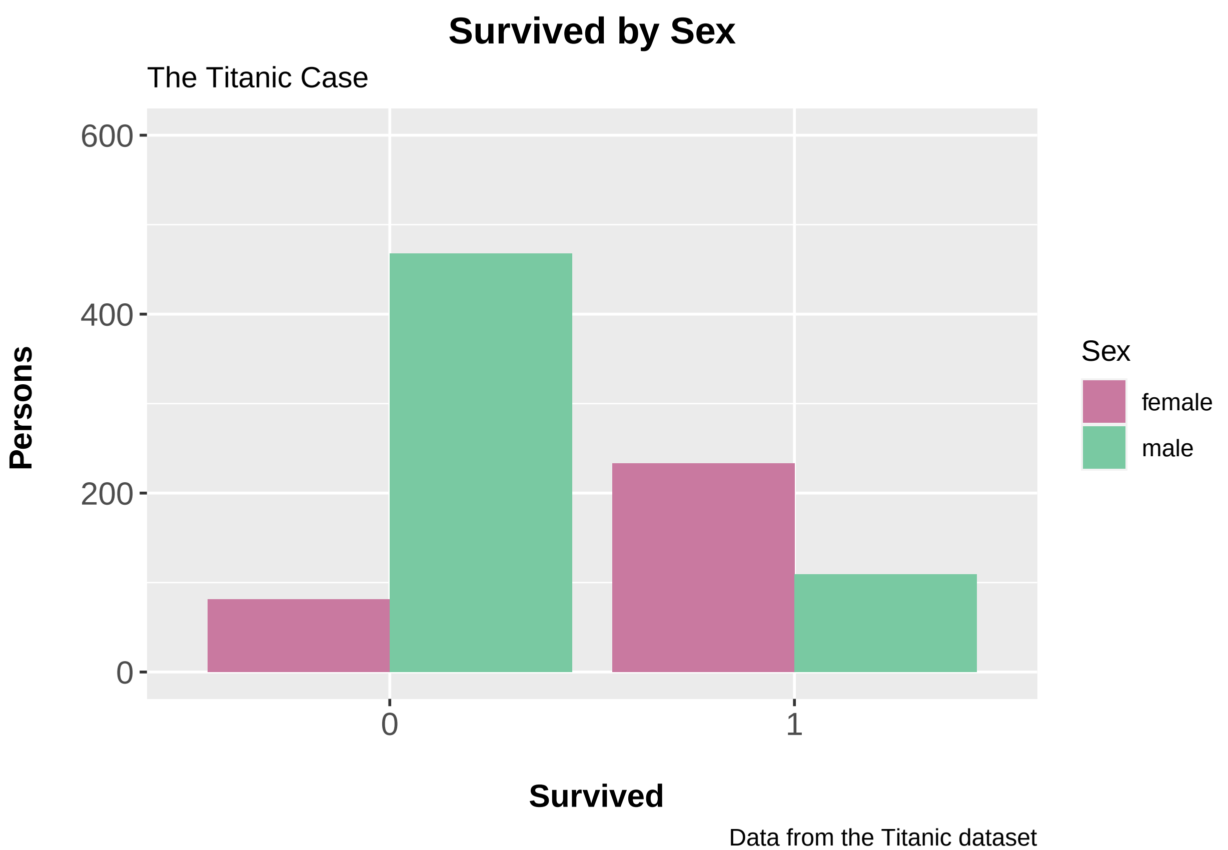
```
## 'data.frame':    1309 obs. of  4 variables:
##  $ Survived: Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass  : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...
##  $ Sex     : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ Embarked: Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
```

Adding some visuals to clarify the picture

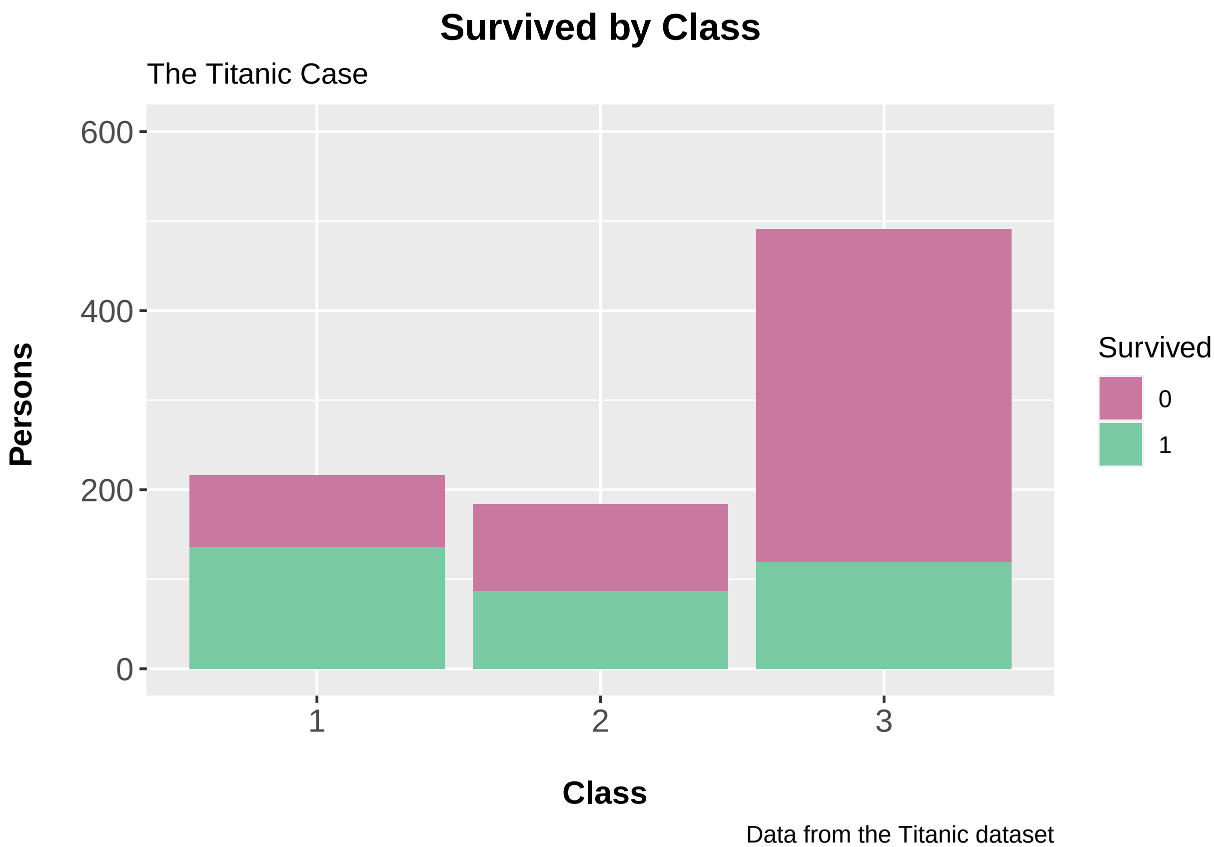
```r
# visualizing the numbers available so to get the general picture of the case
ggplot(data = train_fctrs, aes(x = Survived, fill = Sex)) + geom_bar() +
  scale_y_continuous(limits = c(0, 600)) +              # making visual limits
  scale_fill_manual(values = c("#c979a0", "#79C9A2")) + # color code for sex categories
  labs(title = "Survived by Sex",                       # setting labels
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Survived", y = "Persons \n") +           # placing extra-line for better readability
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```



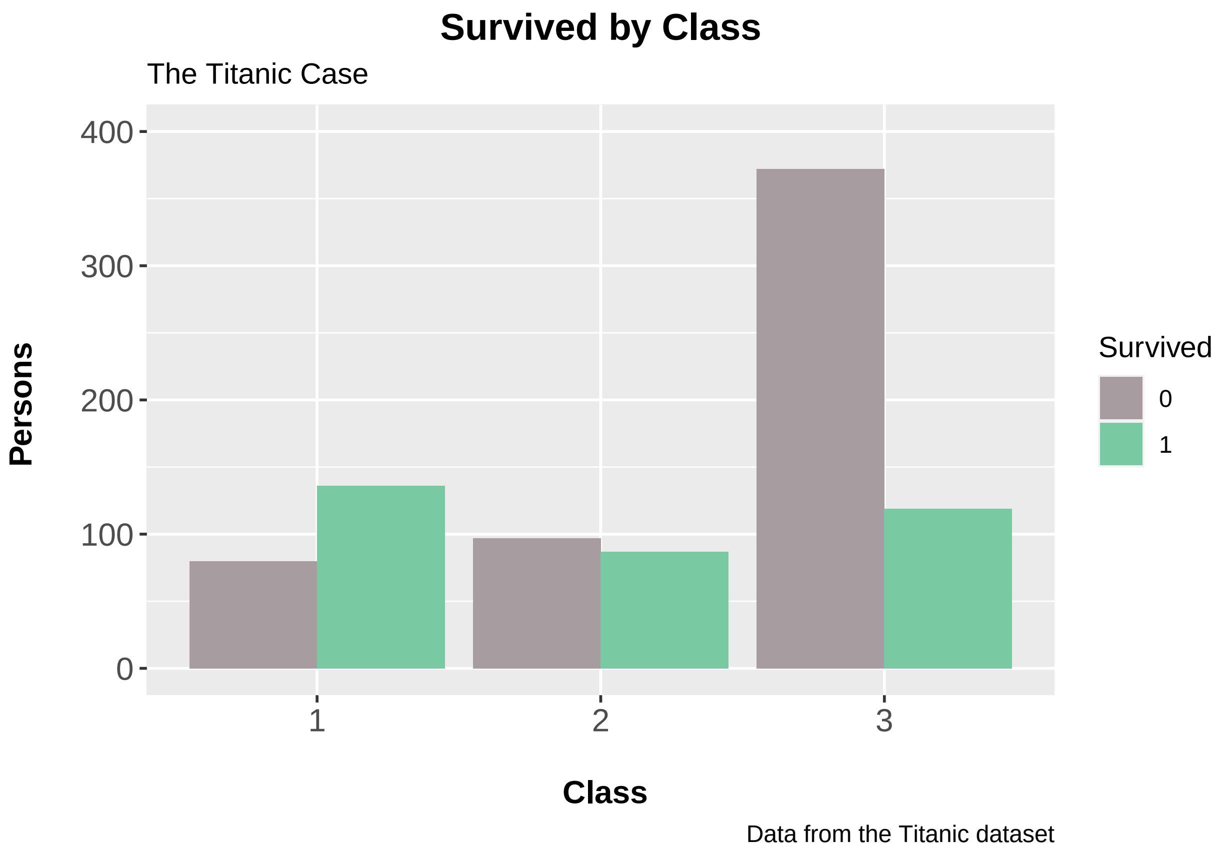
```r
# side-by-side comparison to make things more understandable - Survived by Sex
ggplot(data = train_fctrs, aes(x = Survived, fill = Sex)) +
  geom_bar(position = "dodge") +
  scale_y_continuous(limits = c(0, 600)) +              # making visual limits
  scale_fill_manual(values = c("#c979a0", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Sex",                       # setting labels
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Survived", y = "Persons \n") +
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```




```r
# side-by-side comparison to make things more understandable - Survived by Sex
ggplot(data = train_fctrs, aes(x = Pclass, fill = Survived)) +
  geom_bar() +
  scale_y_continuous(limits = c(0, 600)) +              # making visual limits
  scale_fill_manual(values = c("#c979a0", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Class",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Class", y = "Persons \n")+
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```




```r
# side-by-side comparison to make things more understandable - Survived by Class
ggplot(data = train_fctrs, aes(x = Pclass, fill = Survived)) +
  geom_bar(position = "dodge") +
  scale_y_continuous(limits = c(0, 400)) +
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Class",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Class", y = "Persons \n")+
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```



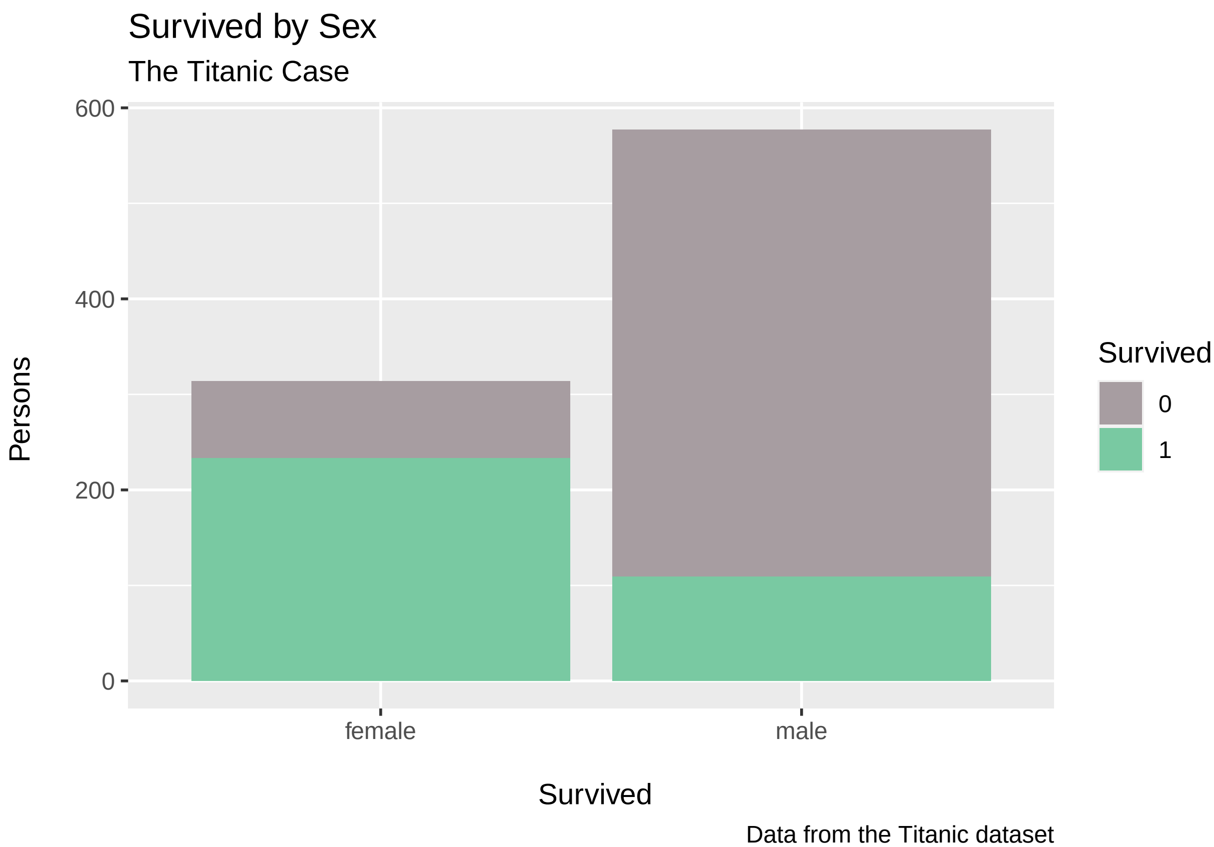

```
# create a train-limited dataset, so the data from train is only considered in calculations

LT = dim(train)[1]
MT = dim(test)[1]  # [1] stands for number of rows, [2] stands for number of columns within the data frame
LT  # train dataset limited - gives number of rows to be used for certain purposes, like limited calculations
```

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

```
MT  # test dataset limited - ...
```

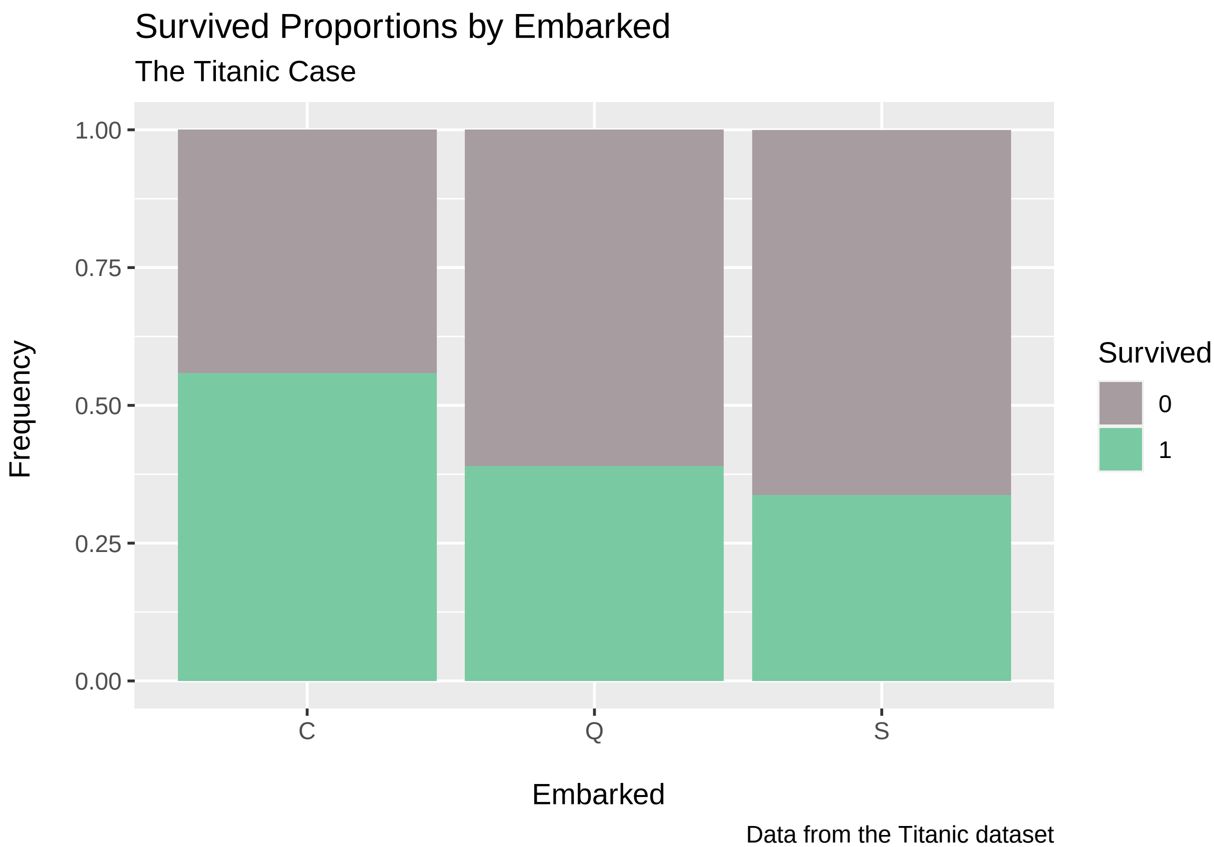
```
## [1] 418
```

```
# check the relationship between Sex and Survival:
ggplot(data = full[1:LT,],aes(x = Sex,fill = Survived)) + geom_bar() + # [1:LT] = look into train data, witho
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Sex",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Survived", y = "Persons \n")
```



Survival as a function of Embarked:

```
ggplot(data = full[1:LT,], aes(x = Embarked, fill = Survived)) +
  geom_bar(position = "fill") +
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived Proportions by Embarked",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Embarked", y = "Frequency \n")
```




```
# get the numbers of Survived within Embarked classes, =1 - survived, =0 - didn't survive
t<-table(full[1:LT,]$Embarked,full[1:LT,]$Survived)
pander(t)                                                                     # pander trick
```

	0	1
C	75	95
Q	47	30
S	427	217

```
pander(addmargins(table(full[1:LT,]$Embarked,full[1:LT,]$Survived))) # pander trick
```

	0	1	Sum
C	75	95	170
Q	47	30	77
S	427	217	644
Sum	549	342	891

```
# get the percentage of Survived within Embarked classes, =1 - survived, =0 - didn't survive
t<-table(full[1:LT,]$Embarked,full[1:LT,]$Survived)
for (i in 1:dim(t)[1]){
  t[i,]<-t[i,]/sum(t[i,])*100
}
pander(t)
```

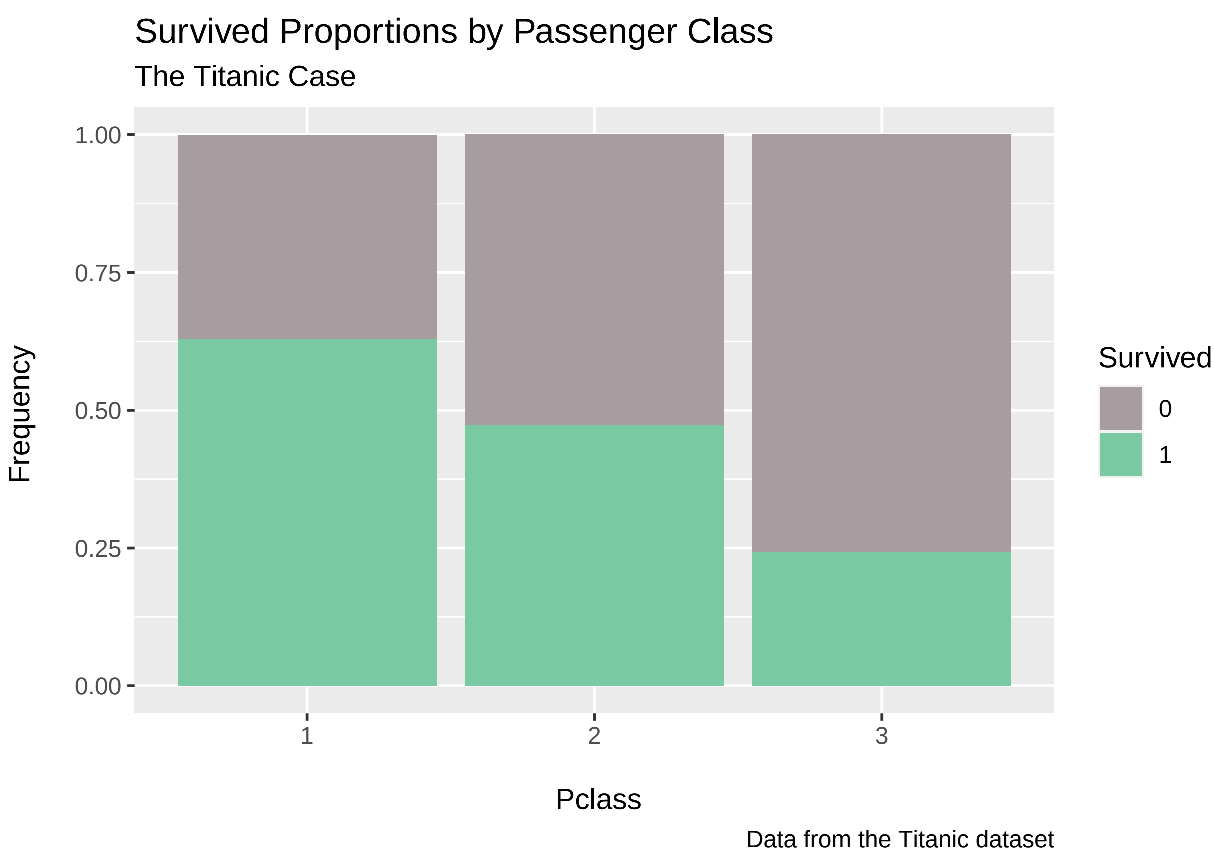
	0	1
C	44.12	55.88
Q	61.04	38.96
S	66.3	33.7

It looks like chances for survival were higher for those Embarked in 'C' (55% compared to 33% and 38%, row-wise).

But it is a bit skewed, if to compare the number of victims and produce column-wise ratio calculations

Survival as a function of Pclass:

```
ggplot(data = full[1:LT, ], aes(x = Pclass, fill = Survived)) +
  geom_bar(position = "fill") +
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived Proportions by Passenger Class",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Pclass", y = "Frequency \n")
```



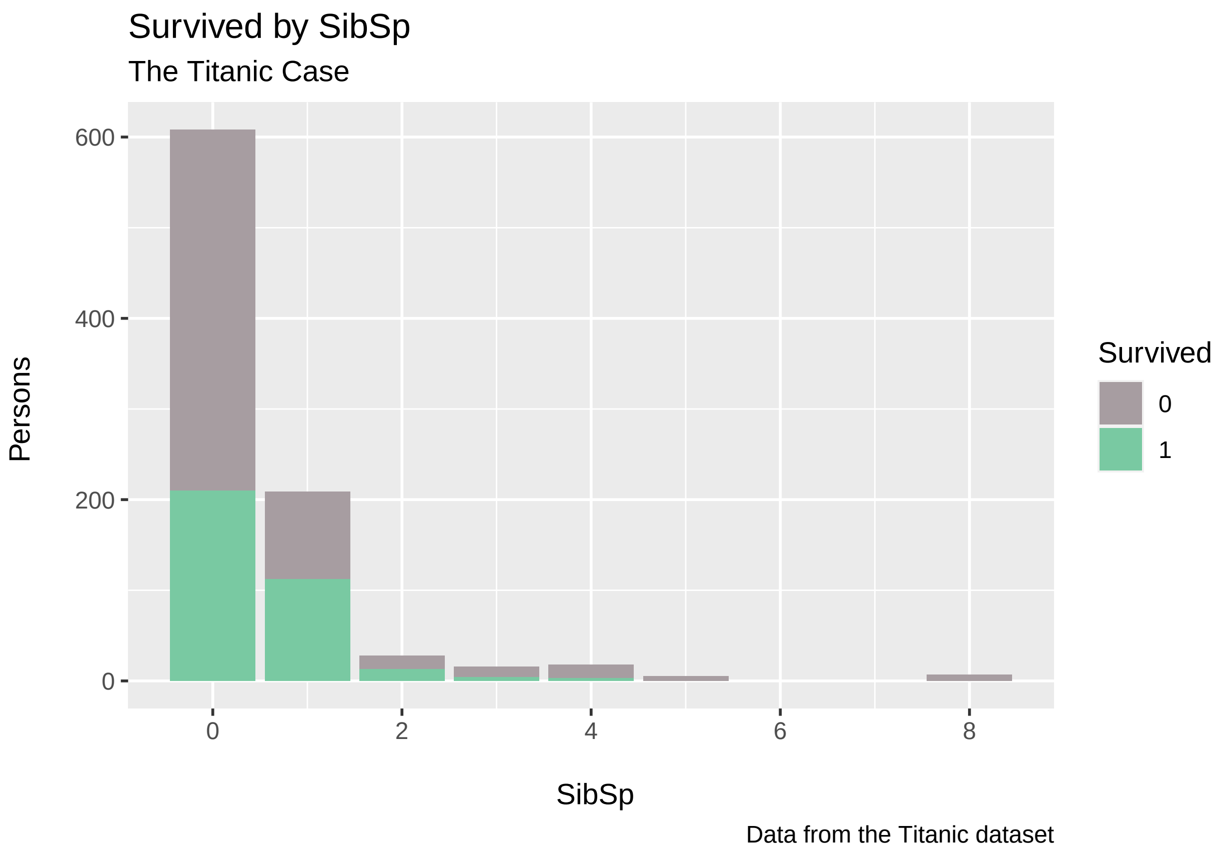
It looks like you have a better chance to survive if you were in lower ticket class.

```
# check the of Embarked versus Pclass:
ggplot(data = full[1:LT,],aes(x=Embarked,fill=Survived))+geom_bar(position="fill") +
  facet_wrap(~Pclass) +
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived Proportions by Passenger Class vs Embarked Type",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Embarked", y = "Frequency \n")
```


```
# Now it's not so clear that there is a correlation between Embarked and Survival.
```

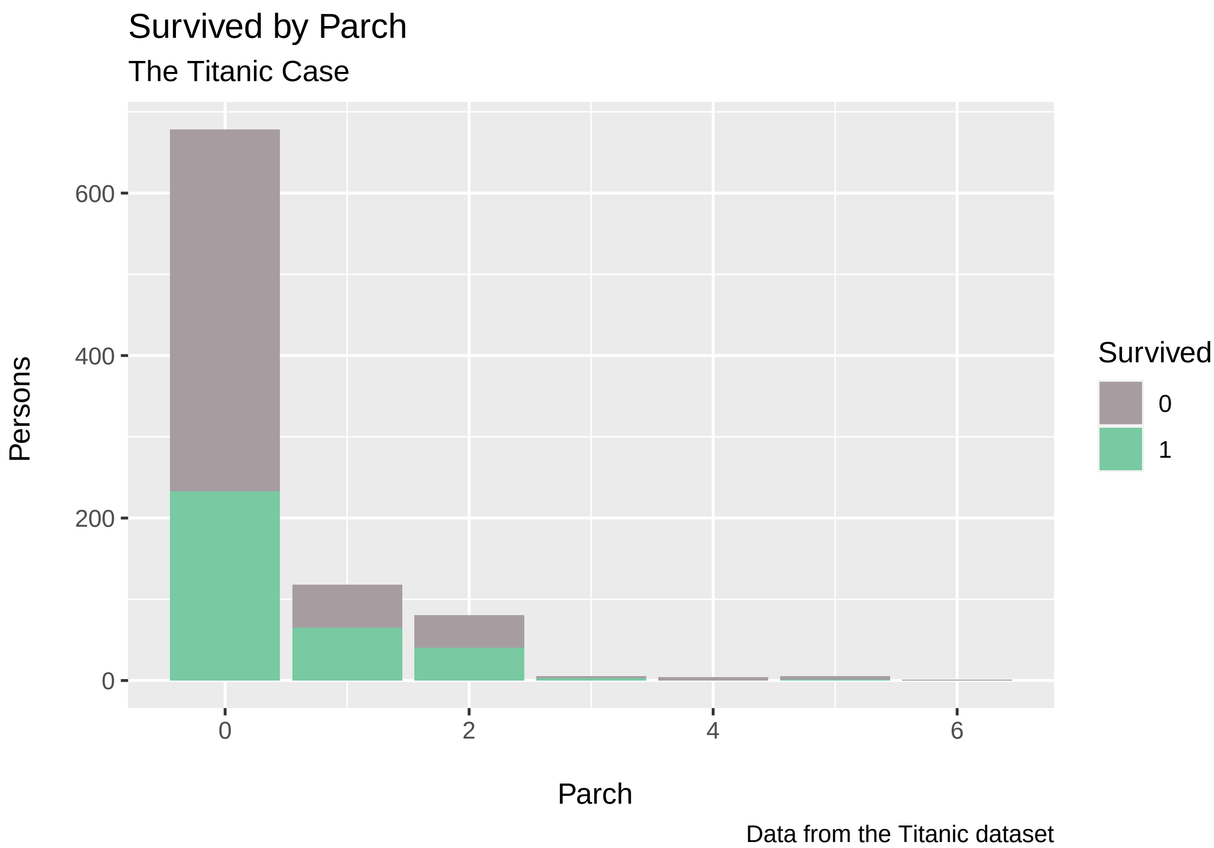
Survival as a function of SibSp

```
ggplot(data = full[1:LT,],aes(x=SibSp,fill=Survived))+geom_bar()+
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by SibSp",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n SibSp", y = "Persons \n")
```



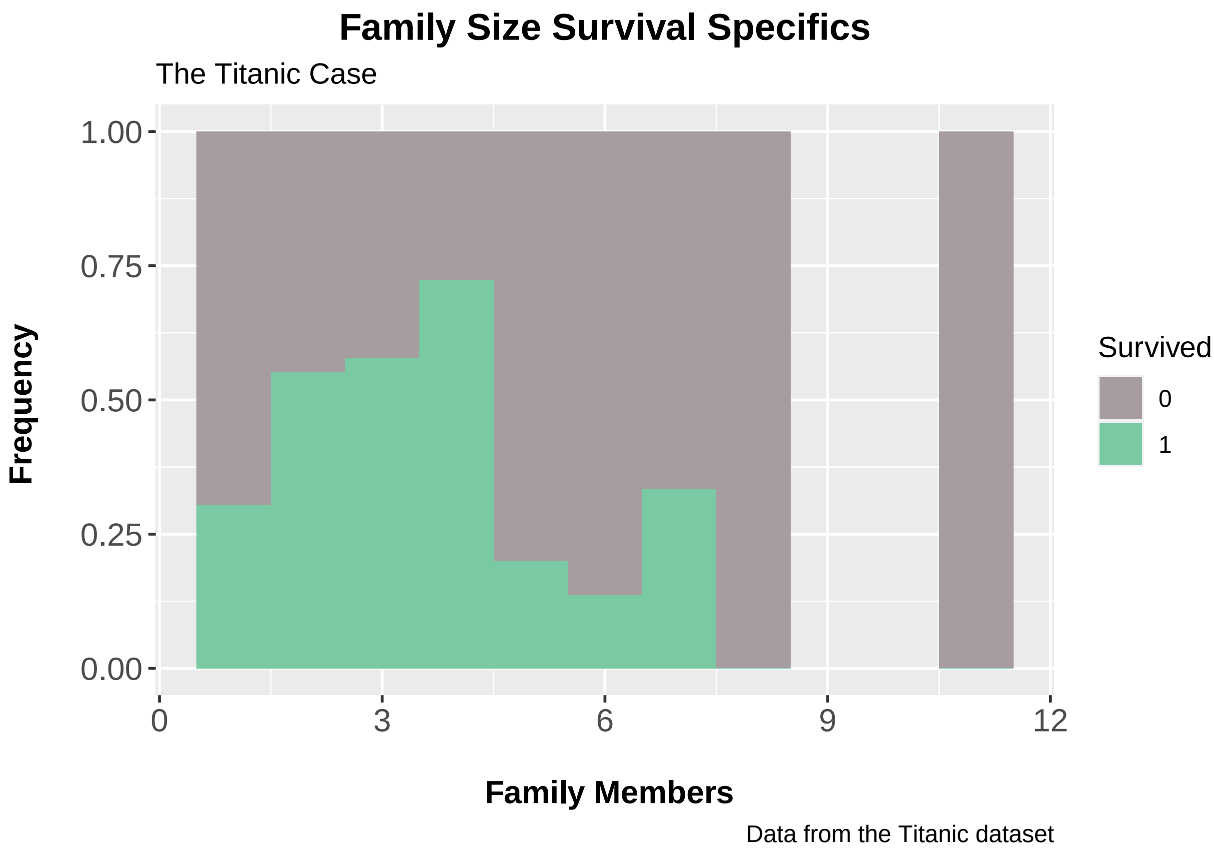
Survival as a function of Parch

```
ggplot(data = full[1:LT,],aes(x=Parch,fill=Survived))+geom_bar()+
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Parch",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Parch", y = "Persons \n")
```




```
# the dynamics of both attributes - SibSp and Parch - seem to be quite similar
```

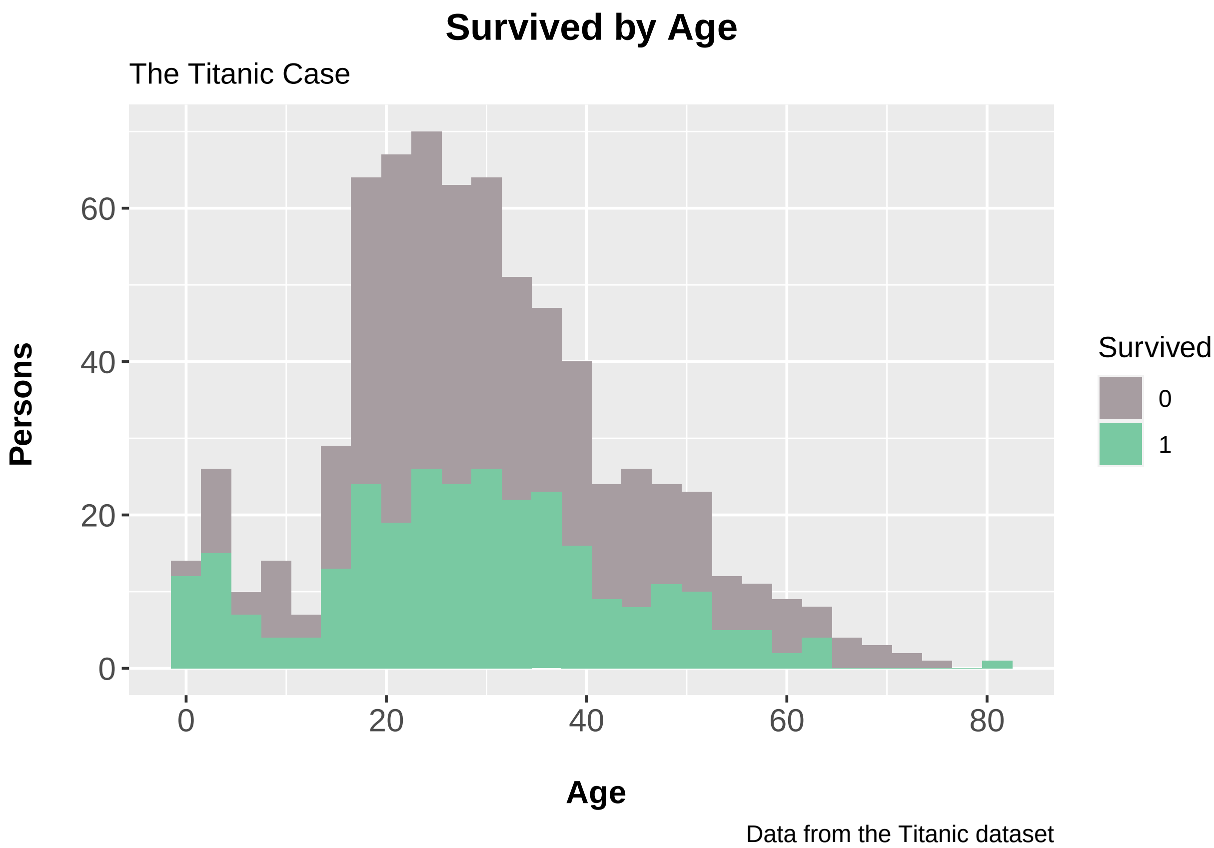
```
# check the attribute of family size.
full$FamilySize <- full$SibSp + full$Parch +1;
full1<-full[1:LT,]
ggplot(data = full1[!is.na(full[1:LT,]$FamilySize),], aes(x=FamilySize,fill=Survived)) +
  geom_histogram(binwidth =1,position="fill")+
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +    # color code for sex categories
  labs(title = "Family Size Survival Specifics",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Family Members", y = "Frequency \n")+
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```




```
# That shows that families with a family size bigger or equal to 2 but less than 6 have a more than 50% to su
# in contrast to families with 1 member or more than 5 members.
```

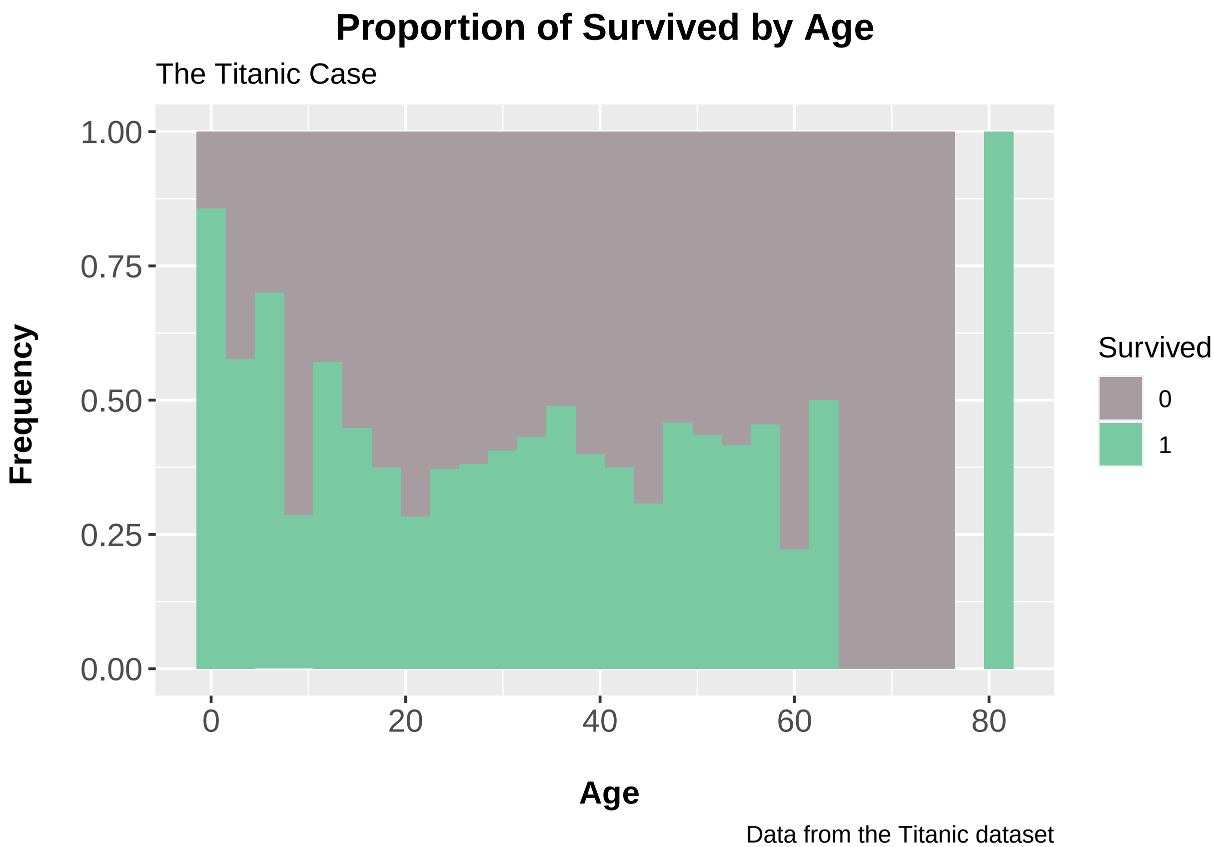
Survival as a function of Age:

```
ggplot(data = full1[!(is.na(full[1:LT,]$Age)),],aes(x=Age,fill=Survived))+geom_histogram(binwidth =3)+
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Age",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Age", y = "Persons \n")+
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```



Survival ratio by Age

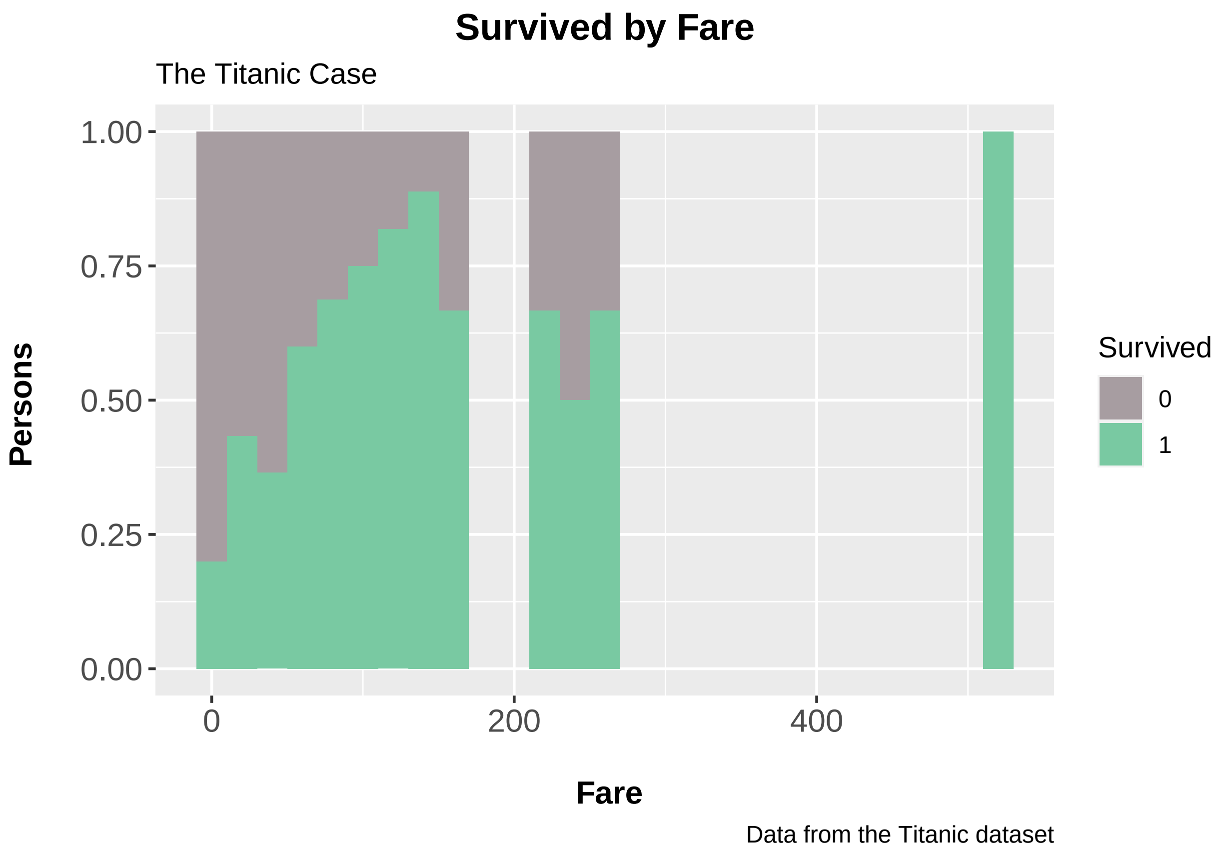
```
ggplot(data = full1[!is.na(full[1:LT,]$Age),],aes(x=Age,fill=Survived))+geom_histogram(binwidth = 3,position=
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +
  labs(title = "Proportion of Survived by Age",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Age", y = "Frequency \n")+
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```




```
# Children (under 15) and old people (80+) had a better chance to survive.
```

Additional research

```
# check the correlation of Fare versus Survivial
ggplot(data = full[1:LT,],aes(x=Fare,fill=Survived))+geom_histogram(binwidth =20, position="fill")+
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Fare",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Fare", y = "Persons \n")+
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```




```
full$Fare[is.na(full$Fare)] <- mean(full$Fare,na.rm=T)
sum(is.na(full$Fare))
```

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

```
# seems like bigger fare gave better chance to survive
```

```
# check the missing values for Age
sum(is.na(full$Age))
```

```
## [1] 263
```

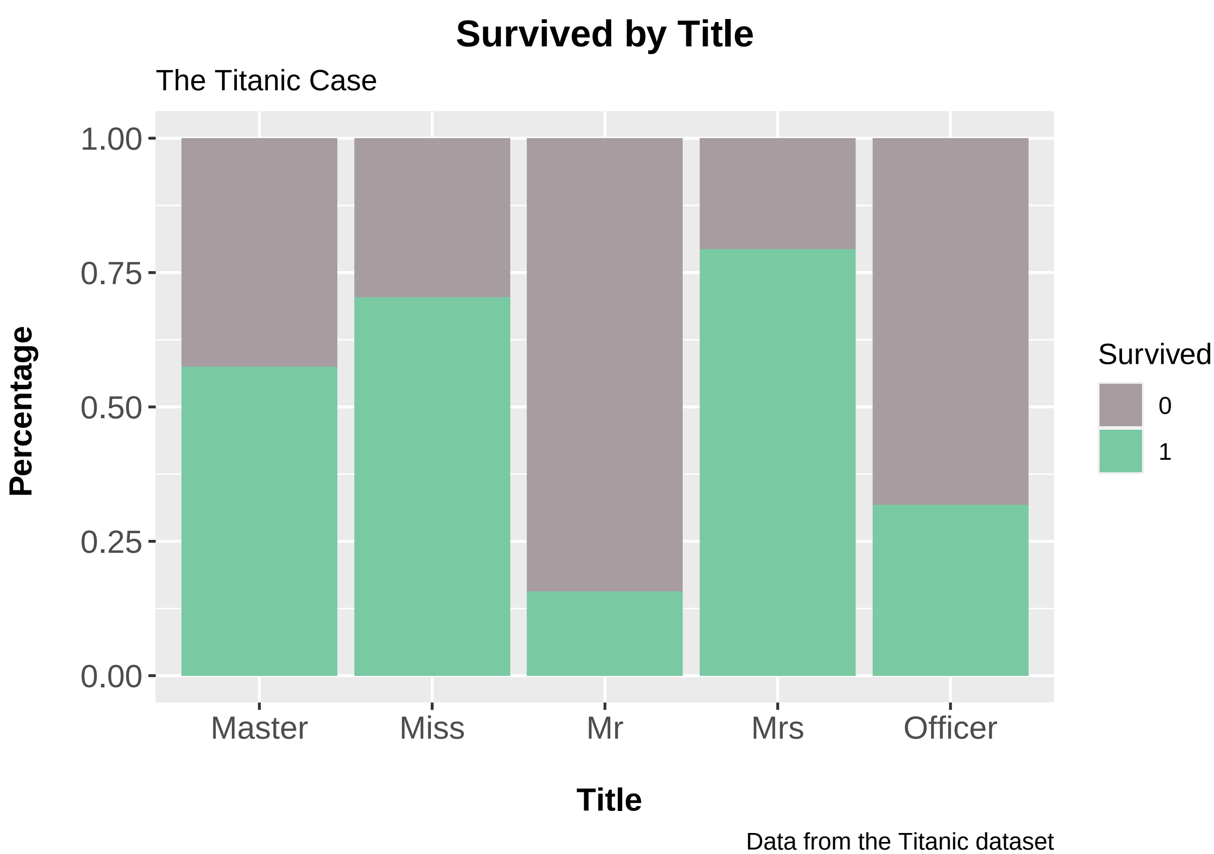
```
# There are a lot of missing values in the Age attribute, put the mean instead of the missing values
full$Age[is.na(full$Age)] <- mean(full$Age,na.rm=T)
sum(is.na(full$Age))
```

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

```
# check the influence of a certain title of a passenger on the survival fact

full$Title <- gsub('(.*, )|(\\..*)', '', full$Name)
full$Title[full$Title == 'Mlle']<- 'Miss'
full$Title[full$Title == 'Ms']<- 'Miss'
full$Title[full$Title == 'Mme']<- 'Mrs'
full$Title[full$Title == 'Lady']<- 'Miss'
full$Title[full$Title == 'Dona']<- 'Miss'
officer<- c('Capt','Col','Don','Dr','Jonkheer','Major','Rev','Sir','the Countess')
full$Title[full$Title %in% officer]<-'Officer'
```

```r
# convert Title into a factor
full$Title<- as.factor(full$Title)
```

```r
# visualize the percentage of Survived vs Title
ggplot(data = full[1:LT,],aes(x=Title,fill=Survived))+geom_bar(position="fill")+
  scale_fill_manual(values = c("#a79da1", "#79C9A2")) +   # color code for sex categories
  labs(title = "Survived by Title",
       subtitle = "The Titanic Case",
       caption = "Data from the Titanic dataset",
       x = "\n Title", y = "Percentage \n")+
  theme(axis.text = element_text(size = 12),
        axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(size = 14, hjust = 0.5, face = "bold"))
```



Prediction

At this time point, let's **predict the chance of survival as a function of the other attributes**. Let's keep just the correlated features: **Pclass, Sex, Age, SibSp, Parch, Title and Fare.**

The *train* dataset will be divided into two sets: training set (*train1*) and test set (*train2*) to be able to estimate the error of the prediction.

```r
# The train set with the important features
train_im<- full[1:LT,c("Survived","Pclass","Sex","Age","Fare","SibSp","Parch","Title")]
ind<-sample(1:dim(train_im)[1],500) # Sample of 500 out of 891
train1<-train_im[ind,]              # The train set of the model
train2<-train_im[-ind,]             # The test set of the model
```

```r
# run a logistic regression
model <- glm(Survived ~.,family=binomial(link='logit'),data=train1)
summary(model)
```

```
##
## Call:
## glm(formula = Survived ~ ., family = binomial(link = "logit"),
##     data = train1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2891  -0.5368  -0.3876   0.5451   2.4751
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.916e+01  8.827e+02   0.022 0.982687
## Pclass2      -1.469e+00  4.454e-01  -3.299 0.000970 ***
## Pclass3      -2.310e+00  4.463e-01  -5.175 2.27e-07 ***
## Sexmale      -1.466e+01  8.827e+02  -0.017 0.986748
## Age          -3.813e-02  1.258e-02  -3.032 0.002430 **
## Fare          2.359e-04  4.299e-03   0.055 0.956249
## SibSp        -6.255e-01  1.798e-01  -3.478 0.000505 ***
## Parch        -3.742e-01  1.755e-01  -2.132 0.033025 *
## TitleMiss    -1.520e+01  8.827e+02  -0.017 0.986265
## TitleMr      -3.440e+00  7.517e-01  -4.577 4.72e-06 ***
## TitleMrs     -1.440e+01  8.827e+02  -0.016 0.986986
## TitleOfficer -3.352e+00  1.069e+00  -3.136 0.001714 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 651.08  on 499  degrees of freedom
## Residual deviance: 407.47  on 488  degrees of freedom
## AIC: 431.47
##
## Number of Fisher Scoring iterations: 13
```

```r
# Let's look at the prediction of this model on the test set (train2):
pred.train <- predict(model,train2)
pred.train <- ifelse(pred.train > 0.5,1,0)
```

```r
# Mean of the true prediction
mean(pred.train==train2$Survived)
```

It results as attributes SibSp, Parch and Fare are not statisticaly significant.

```
## [1] 0.831202
```

```r
# make a summary table of the prediction model
t1<-table(pred.train,train2$Survived)
pander(t1)
```

	0	1
0	217	56
1	10	108

```r
# Presicion and recall of the model
presicion<- t1[1,1]/(sum(t1[1,]))
recall<- t1[1,1]/(sum(t1[,1]))
# get the precision and recall parameters
presicion
```

```
## [1] 0.7948718
```

```r
recall
```

```
## [1] 0.9559471
```

```r
# F1 score
F1<- 2*presicion*recall/(presicion+recall)
F1
```

```
## [1] 0.868
```

```r
# Let's run it on the test set:
test_im<-full[LT+1:1309,c("Pclass","Sex","Age","SibSp","Parch","Fare","Title")]
```

```r
#  make at the prediction of this model on the test set:
pred.test <- predict(model,test_im)[1:418]
pred.test <- ifelse(pred.test > 0.5,1,0)
```

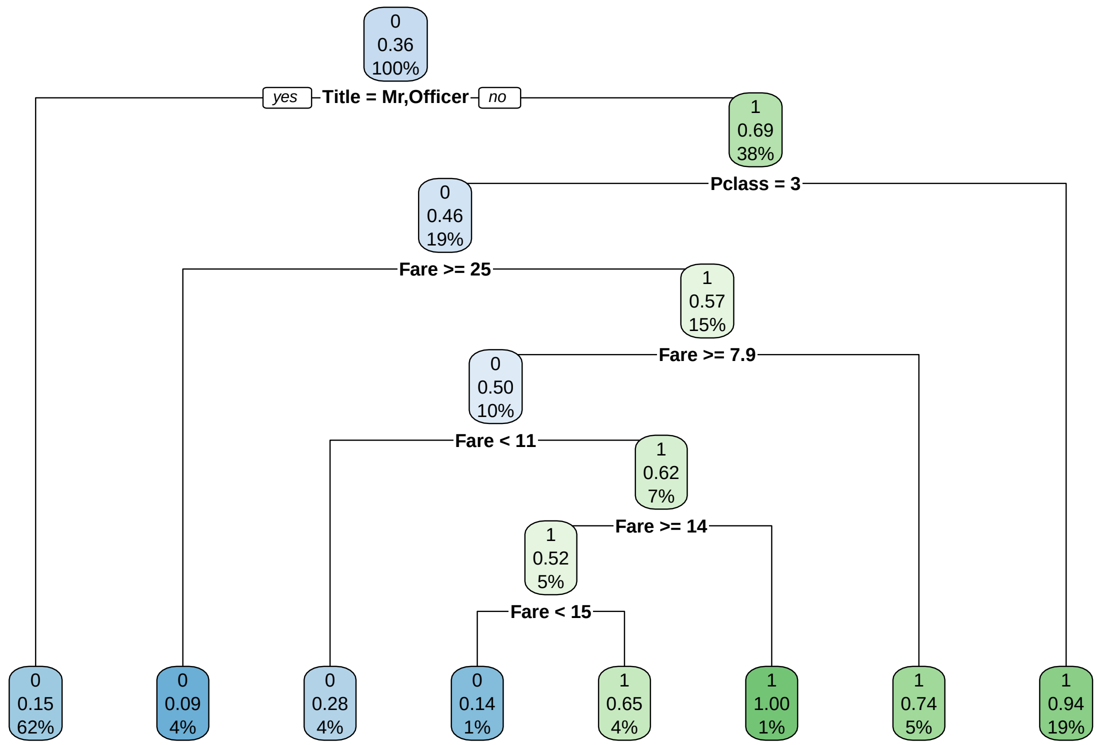
```r
# put result into a data frame
res<- data.frame(test$PassengerId,pred.test)
names(res)<-c("PassengerId","Survived")
```

```r
# put the prediction result into a .csv file
write.csv(res,file="prediction.csv",row.names = F)
```

F1 score on the initial test resulted at 0.879. That's pretty good

Building a tree...

```r
# plant a tree and visualize it
model_dt<- rpart(Survived ~.,data=train1, method="class")
rpart.plot(model_dt)
```




```
# make the prediction on the model
pred.train.dt <- predict(model_dt,train2,type = "class")
mean(pred.train.dt==train2$Survived)
```

```
## [1] 0.8337596
```

```
t2<-table(pred.train.dt,train2$Survived)

presicion_dt<- t2[1,1]/(sum(t2[1,]))
recall_dt<- t2[1,1]/(sum(t2[,1]))
```

```
# get the precision and recall
presicion_dt
```

```
## [1] 0.8091603
```

```
recall_dt
```

```
## [1] 0.9339207
```

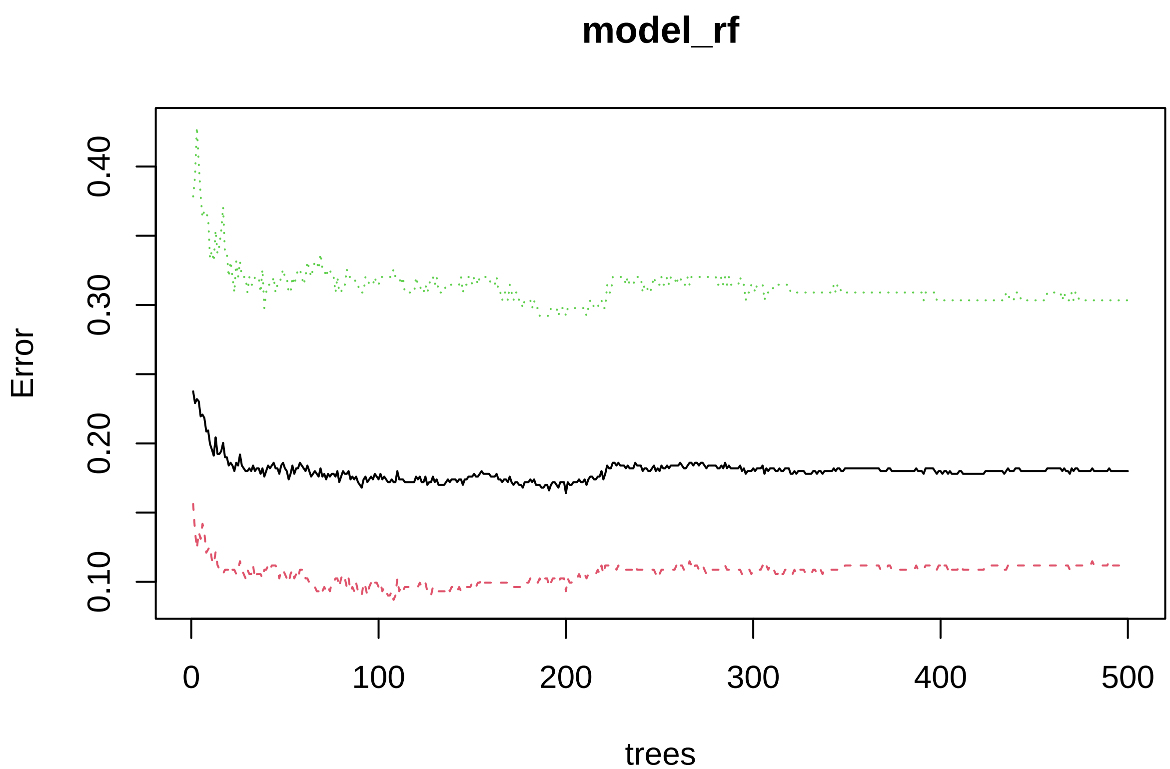
```
# get the F1 score
F1_dt<- 2*presicion_dt*recall_dt/(presicion_dt+recall_dt)
F1_dt
```

```
## [1] 0.8670757
```

```r
# run this model on the test set:
pred.test.dt <- predict(model_dt,test_im,type="class")[1:418]
res_dt<- data.frame(test$PassengerId,pred.test.dt)
names(res_dt)<-c("PassengerId","Survived")
write.csv(res_dt,file="prediction_dt.csv",row.names = F)
```

```r
# make prediction on survival using a random forest
model_rf<-randomForest(Survived~.,data=train1)
```

```r
# Let's look at the error
plot(model_rf)
```




```r
# make the prediction on the model
pred.train.rf <- predict(model_rf,train2)
mean(pred.train.rf==train2$Survived)
```

```
## [1] 0.8337596
```

```r
t1<-table(pred.train.rf,train2$Survived)
presicion<- t1[1,1]/(sum(t1[1,]))
recall<- t1[1,1]/(sum(t1[,1]))
presicion
```

```
## [1] 0.8139535
```

```r
recall
```

```
## [1] 0.9251101
```

```
F1<- 2*presicion*recall/(presicion+recall)
F1
```

```
## [1] 0.8659794
```

```
# Let's run this model on the test set:
pred.test.rf <- predict(model_rf,test_im)[1:418]
res_rf<- data.frame(test$PassengerId,pred.test.rf)
names(res_rf)<-c("PassengerId","Survived")
write.csv(res_rf,file="submission_rf.csv",row.names = F)
```

Conclusion

The mean of the right predictions:

- decision tree method: 0.77837
- random forest method: 0.77990
- logistic regression model: 0.7488

---

updated:

- 04/04/2023 - Made some cosmetics on some visuals (labels, theme…)

The colors picked up considering some wide-spread advice on visualization within the industry - to select the complementary colors for better readability.

Used this resourse for the colour selection, actually.

- 09/04/2023 - Including pander library for certain table re-shaping