Predictind Survival on the Titanic using Different Classifiers.

Dennis Murithi

Introduction

The different machine learning algorithms to be used will be :

- **Random Forest**
- **Logistic Regression**, and
- **Naive Bayes**

Loading the required packages

```
library(tidyverse)
library(ggthemes)
library(corrplot)
library(VIM)
library(caret) # machine learning
library(RANN) # knnInpute
library(reshape2)
```

Raeding in the data

```
train_data = read.csv("C:/Users/adm/Documents/Datasets/titanic survival/train.csv",na.strings = "")
test_data = read.csv("C:/Users/adm/Documents/Datasets/titanic survival/test.csv",na.strings = "")

full_data <- bind_rows(train_data,test_data)
```

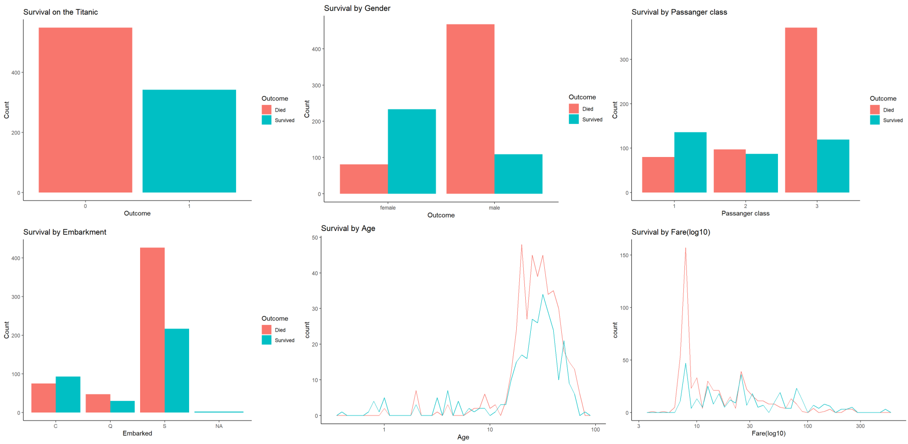
Checking at the first data rows

```
head(full_data)
```

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

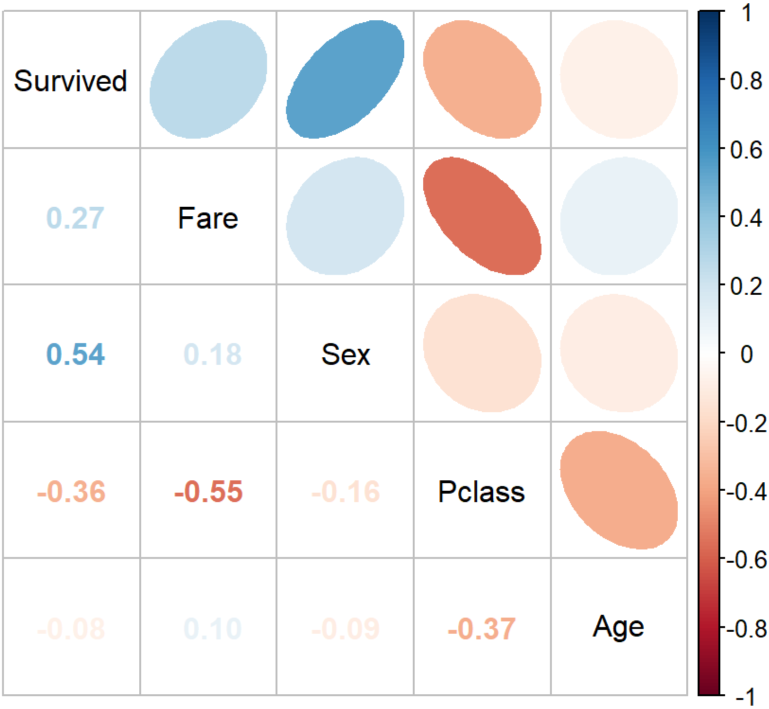
Exploratory Data Analysis

Graphing some of the variables to see how they affect survival rate.



From these graphs we can gather that: * Most of the passengers on the Titanic died. * Women had a better chance of survival than men with the majority of them surviving and the men died. * Those who embarked at *C* had a slightly higher chance of survival than those who embarked at other places. * There seems to be a trend of those younger than 16 having a higher chance of survival than death. * Passengers who paid a higher fare had a higher survival chance than those that paid less.

Checking the correlation between these variables and survival to get a better overview of the importance.



Out of these variables we can see that sex, passenger class, followed by fare have a small to median correlation with survival; in addition they might be important in predicting survival.

Feature Engineering

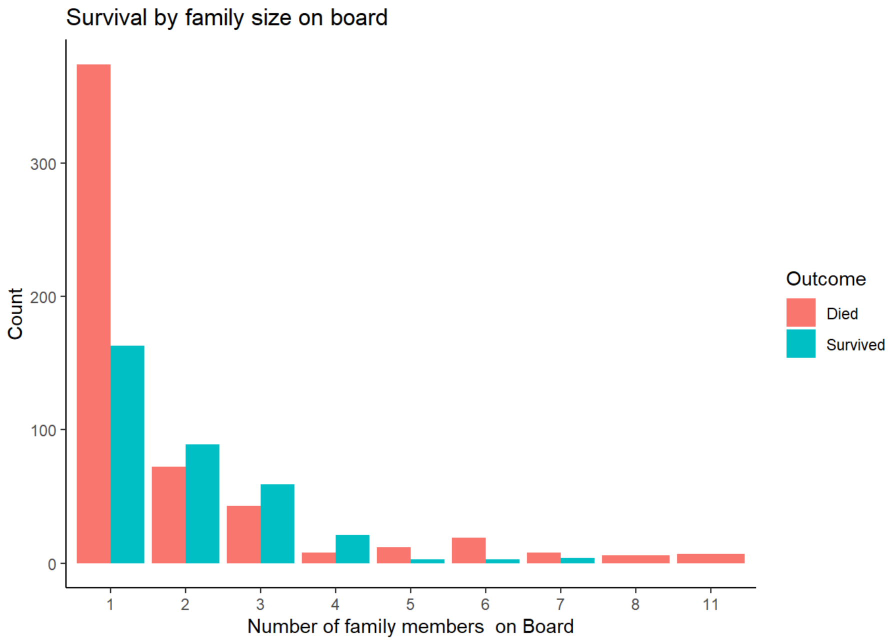
Doing feature engineering to create some new variables : (1) **family_size**, (2)**Title**, (3)**Cabin_letter**, and (4)**Ticket_Number**.

Family Size

To create the *family_size* variable we will add *Sibsp* (siblings+spouse) with *Parch* (Parents + children) plus1 (for the individuals themselves).

```
full_data$family_size = full_data$SibSp+full_data$Parch+
        1
```

plotting this new variable to see how it is like

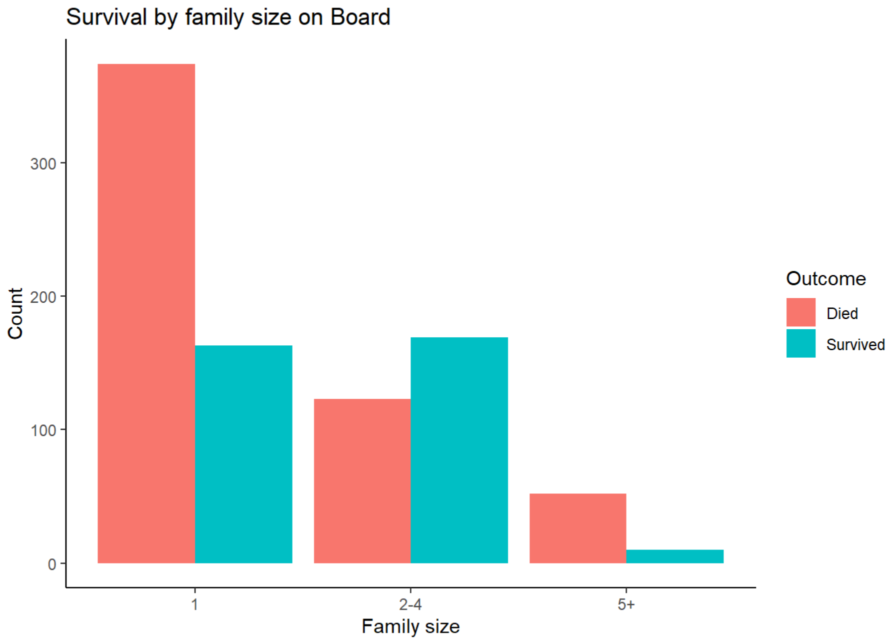


Looks like it was beneficial to have a family size of 2-4 members on board while those alone or in families of 5 and larger had lower chances of survival.Creating a new variable that shows these divisions as it might improve our predictive model.

```
# Create categories for family size: 1, 2-4, 5+
full_data$family_size_range = cut(full_data$family_size,
        c(0, 1, 4, 15), include.lowest = TRUE)

# Next, fix the names of the variables.
levels(full_data$family_size_range) = c('1', '2-4', '5+'
        )
```

Plotting this new variable to see how it compares to *family_size* variable.



Here we can clearly see that with a family size of 2-4 you had a better survival chance than those who were alone or with family members of 5 or more on board.

Title

Engineering the *Title* variable ,by extracting the title from the existing the existing *Name* variable.

Using regular expression to extract the variable.

```
full_data$Title <- gsub('(.*, )|(\\..*)', '', full_data$
         Name)
```

Take a look at a table of the titles.

```
table(full_data$Title)
```

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

Since some titles have few occurrence reassign these to a new category *rare_title*.

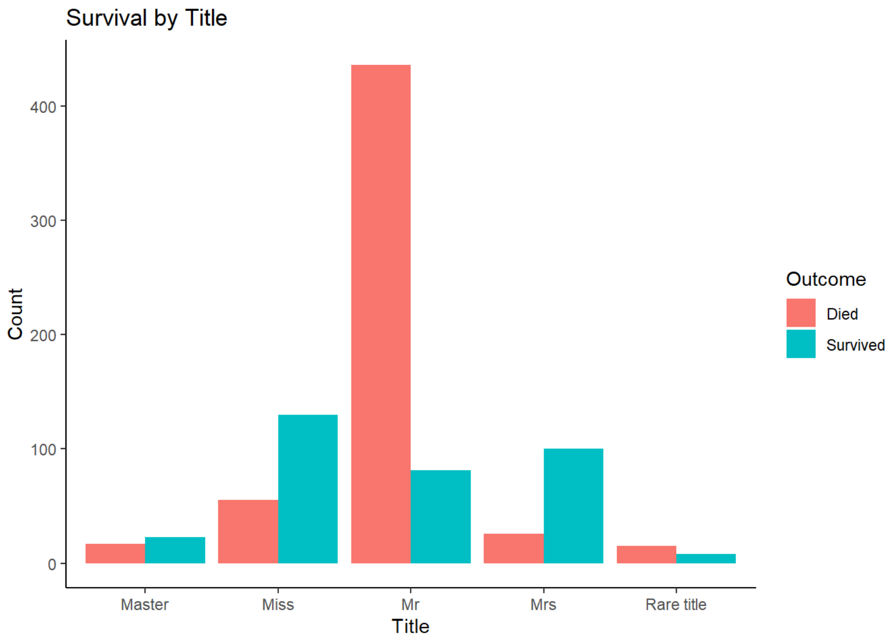
```
rare_title = c('Capt', 'Col', 'Don', 'Jonkheer', 'Lady',
         'Major', 'Rev', 'Sir',
               'the Countess', 'Dr')

full_data$Title[full_data$Title %in% rare_title] <- 'Rar
         e title'
```

Reassign some of the titles to appropriate categories.

```
full_data$Title[full_data$Title=="Mlle"]<-"Miss"
full_data$Title[full_data$Title=="Ms"]<-"Miss"
full_data$Title[full_data$Title=="Dona"]<-"Miss"
full_data$Title[full_data$Title=="Mme"]<-"Mrs"
```

Plotting the Title to see how it affects survival rate.



Here its clear that those with the title Miss or Mrs were the best off in-terms of survival.This is in line with what one would expect since survival rate was higher among women.Those with the title Master seems they had a higher chance of survival.The title Mr shows a clear trend towards death.

Cabin

Extracting the cabin letter from the *Cabin* variable to create *Cabin_letter*.

Using regular expression to extract the cabin letter.

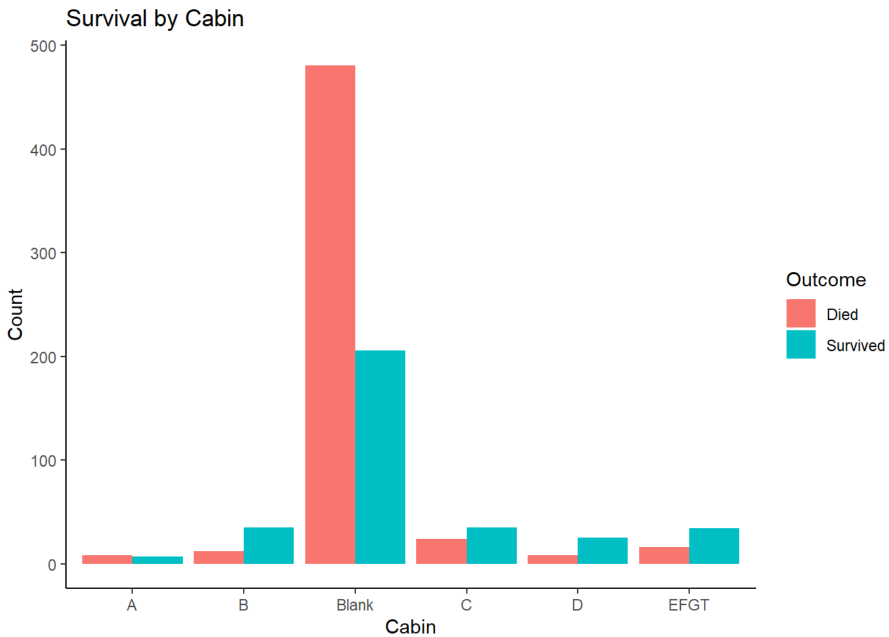
```
full_data$Cabin_letter <- gsub('[0-9].*', '', full_data$
         Cabin)
```

Some cabins have few occurrences and some are categorized under two cabins.Will combine this to create a new cabin indicator called *EFGT*.Also recording those with out cabin letters to *BLANK*.

```
full_data$Cabin_letter[full_data$Cabin_letter=='E'] <-
        'EFGT'
full_data$Cabin_letter[full_data$Cabin_letter=='F'] <-
        'EFGT'
full_data$Cabin_letter[full_data$Cabin_letter=='F E'] <-
        'EFGT'
full_data$Cabin_letter[full_data$Cabin_letter=='F G'] <-
        'EFGT'
full_data$Cabin_letter[full_data$Cabin_letter=='G'] <-
        'EFGT'
full_data$Cabin_letter[full_data$Cabin_letter=='T'] <-
        'EFGT'

full_data$Cabin_letter[is.na(full_data$Cabin_letter)] <-
        'Blank'
```

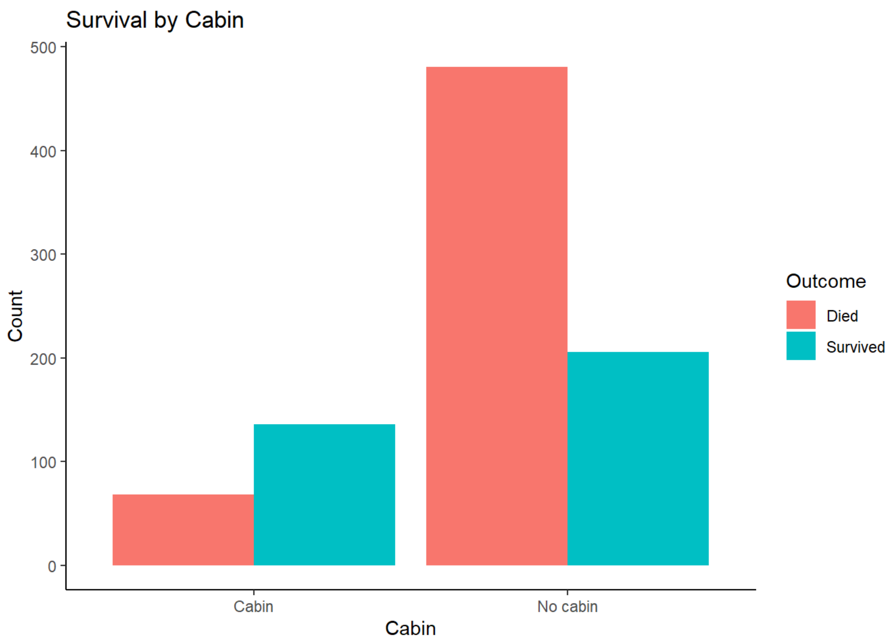
Plotting to see how cabin affects survival.



Looking at this graph , it seems like those who were assigned cabins had a higher chance of survival than those who were not.Creating a new variable that checks cabin presence to look at this in more detail.

```
full_data$cabin_presence[full_data$Cabin_letter=='Blank'
        ] <- 'No cabin'
full_data$cabin_presence[is.na(full_data$cabin_presenc
        e)] <- 'Cabin'
```

Plotting this.



People with a cabin assigned were better off!

Ticket Number

Extracting *Ticket_number* from *Ticket* by removing any non numeric characters using regular expressions.

```
full_data$Ticket_number <- gsub('[^0-9]', '', full_data$
         Ticket)
```

looking at the ticket numbers which have become blank(““) and reassign them.

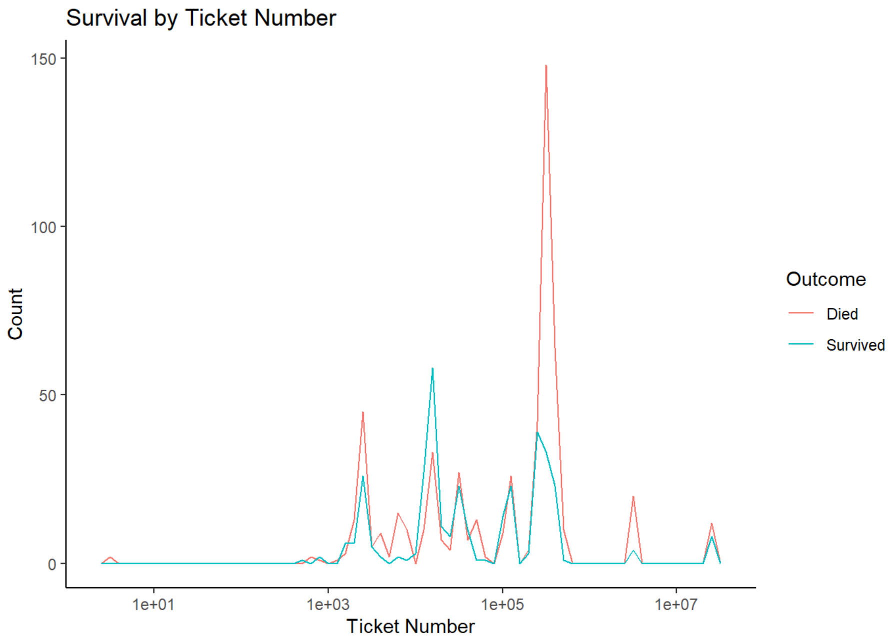
```
table(full_data$Ticket_number=="")

full_data$Ticket_number[full_data$Ticket_number==""] <-
         0
```

```
##
## FALSE  TRUE
##  1305     4
```

Plotting the log of the ticket number.

```
full_data$Ticket_number <- as.integer(full_data$Ticket_n
         umber)
ggplot(full_data[0:891,])+
  geom_freqpoly(aes(x=Ticket_number,color =  factor(Surv
         ived)), binwidth = 0.1)+
  scale_x_log10()+
  scale_color_discrete(name = "Outcome", labels= c("Die
         d","Survived"))+
  theme_classic()+
  labs(title = "Survival by Ticket Number",x = "Ticket N
         umber",y ="Count")
```



There is no immediate trend between ticket number and survival.Checking the correlation between the two variables.

```
cor(full_data$Ticket_number, as.numeric(full_data$Surviv
         ed), use = 'complete.obs')
```

```
## [1] -0.01561505
```

There seems to be no real correlation going on here, this will be dis- cluded from the prediction model.

Fixing the Missing Value

Preparing the data.

Creating a subset of the data and including only relevant variables to use in the prediction model later.

```
full_data_relevant <- subset(full_data, select = c(Survi
        ved, Pclass, Sex, Age, Fare,
                                  Title, cabin_presenc
        e,
                                  family_size_range
                                # Ticket_number, Embark
        ed
                                  ))
```

Making sure each variable is classified as a integer or a factor.

```
full_data_relevant$Survived <- as.factor(full_data_relev
        ant$Survived)
full_data_relevant$Pclass <- factor(full_data_relevant$P
        class, ordered = TRUE)
full_data_relevant$Survived <- as.factor(full_data_relev
        ant$Survived)
full_data_relevant$Title <- as.factor(full_data_relevant
        $Title)
full_data_relevant$cabin_presence <- as.factor(full_data
        _relevant$cabin_presence)
```

count of all unique values in the model sapply(lapply(train, unique), length)

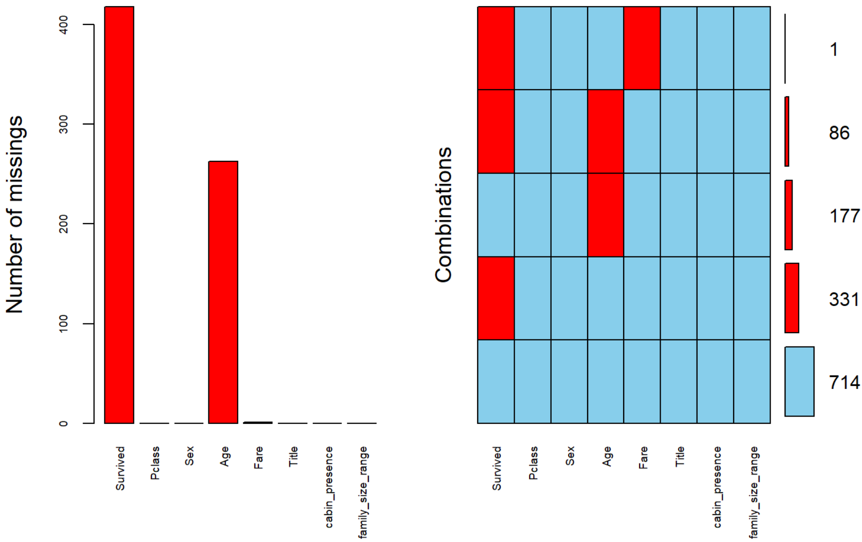
Distribution of missing Values

```
knitr::kable(sapply(full_data_relevant,function(x) sum(i
        s.na(x))))
```

	x
Survived	418
Pclass	0
Sex	0
Age	263
Fare	1
Title	0
cabin_presence	0
family_size_range	0

Treating Missing values for **Fare** and **Age**.

```
aggr(full_data_relevant,sorVars = TRUE, prop = FALSE, ce
         x.axis = .6, numbers = TRUE)
```



Using the `preProcess()` to pre process the missing values model using knnInpute. This will scale the data. Exclude the Survived variable from the pre process model and add it later.

```
#md_prediction <- preProcess(full_data_relevant[c(2:8)],
         method = c('knnImpute',"center","scale"))
md_prediction <- preProcess(full_data_relevant[c(2:8)],
         method = c('knnImpute', 'center', 'scale'))
print(md_prediction)
```

```
## Created from 1045 samples and 7 variables
##
## Pre-processing:
##   - centered (3)
##   - ignored (4)
##   - 5 nearest neighbor imputation (3)
##   - scaled (3)
```

Using the model to predict the missing values that are continuous i.e Fare and Age. NA's for embarked will be computed later.

```
#full_data_complete <- predict(md_prediction,newdata = full_data_relevant[c(2:8)])
full_data_complete <- predict(md_prediction, newdata = full_data_relevant[c(2:8)])
```

Now adding the 'Survived' factor back to the data frame and create a new data frame with full_data_complete and Survived from full_data.

```
full_data_final <- data.frame(full_data_complete,full_data$Survived)
full_data_final <- cbind(full_data$PassengerId,full_data_final)
```

Renaming the `full_data.Survived` column back to `Survived` and turn it back into a factor.

```
full_data_final <- rename(full_data_final,Survived = full_data.Survived,PassangerId = `full_data$PassengerId`)
full_data_final$Survived <- as.factor(full_data_final$Survived)
```

Machine Learning

Splitting the data into train and test for modeling

```
train <- full_data_final[1:891,]
test <- full_data_final[892:1309,]
```

Random Forest

```
set.seed(222)
rf_model <- train(Survived ~ .,method = 'rf',data = trai
         n); print (rf_model)
```

```
## Random Forest
##
## 891 samples
##   8 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 891, 891, 891, 891, 891, 89
1, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.8282402  0.6290053
##    7    0.8215496  0.6164293
##   13    0.8070731  0.5839621
##
## Accuracy was used to select the optimal model using t
he largest value.
## The final value used for the model was mtry = 2.
```

Checking the confusion matrix of the rf model

```
confusionMatrix(rf_model)
```

```
## Bootstrapped (25 reps) Confusion Matrix
##
## (entries are percentual average cell counts across re
samples)
##
##           Reference
## Prediction    0    1
##          0 55.1 10.6
##          1  6.6 27.7
##
##  Accuracy (average) : 0.8281
```

Plotting the model error rate in our prediction

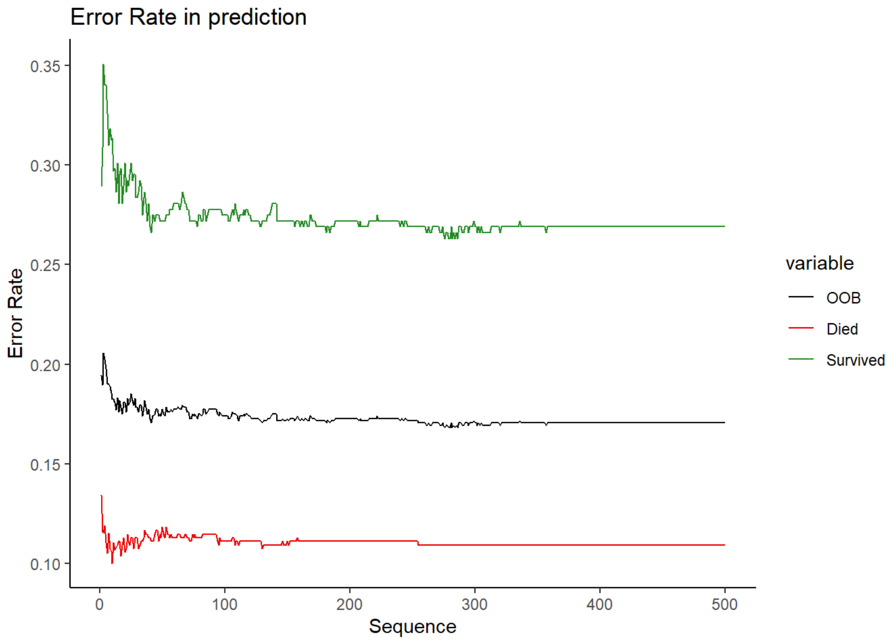
```
# Create data frame of error rate
rf_err_model <- as.data.frame(rf_model[["finalModel"]][[
        "err.rate"]])
rf_err_model$sequence <- seq(1:500)
```

Rename 0's to Died and 1's to Survived

```
rf_err_model <- rename (rf_err_model, Died = '0',Survive
        d = '1')
rf_err_model <- melt(rf_err_model,id = 'sequence')
```

Plotting the error rate

```
ggplot(rf_err_model,aes(x= sequence, y= value, color =
          variable))+
  geom_line()+
  scale_color_manual(values = c("black","red2","forestgr
         een"))+
  theme_classic()+
  labs(title = 'Error Rate in prediction',x = "Sequence"
         ,y= "Error Rate")
```

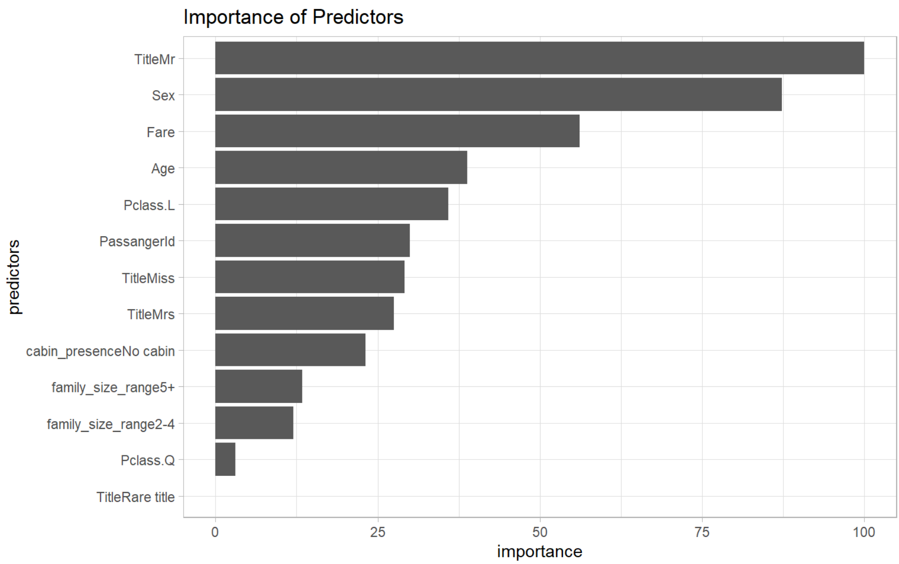


It can be seen the prediction of death is at a higher/ greater accuracy than survival.

Plotting to visualize the variable importance in the prediction.

```
rf_importance <- varImp(rf_model)

ggplot(rf_importance, aes(x=reoreder(variable, importanc
         e),y = importance))+
  geom_bar(stat = "identity")+
  labs(title =  "Importance of Predictors", x= "predicto
         rs", y = "importance")+
  theme_light()
```



Prediction: Random Forest

Predict using the test set.

```
prediction_rf <- predict(rf_model,test)
```

Write the solution to a data frame wit two columns: passengerId and Survived.

```
solution_rf <-data.frame(PassangerID = test$PassangerId,
         Survived =prediction_rf)
```

Write the solution to a file.

```
write.csv(solution_rf,file = 'rf_Titanic_solution.csv',r
         ow.names = FALSE)
```

> **Predicted accuracy of Random Forest model: 82.899%,**
>
> **Leader board accuracy: 79.904%.**

Logistic Regression

Create an object for a 10 fold cross validation. (will be used in the train model)

```
fitControl <- trainControl(method = "cv", number = 10,sa
         vePredictions = TRUE)
```

Creating a predictor model wit `train()` , specifying `method = 'glm'` and `family = binomial()` for the logistic regression.

```
set.seed(222)
lr_model <- train(factor(Survived) ~ .,
                 data =  train,
                 method = 'glm',
                 family = binomial(),
                 trControl = fitControl);print(lr_model)
```

```
## Generalized Linear Model
##
## 891 samples
##   8 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 802, 801, 802, 802, 802, 80
2, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8271411  0.6290895
```

Check the accuracy of the logistic regression model.

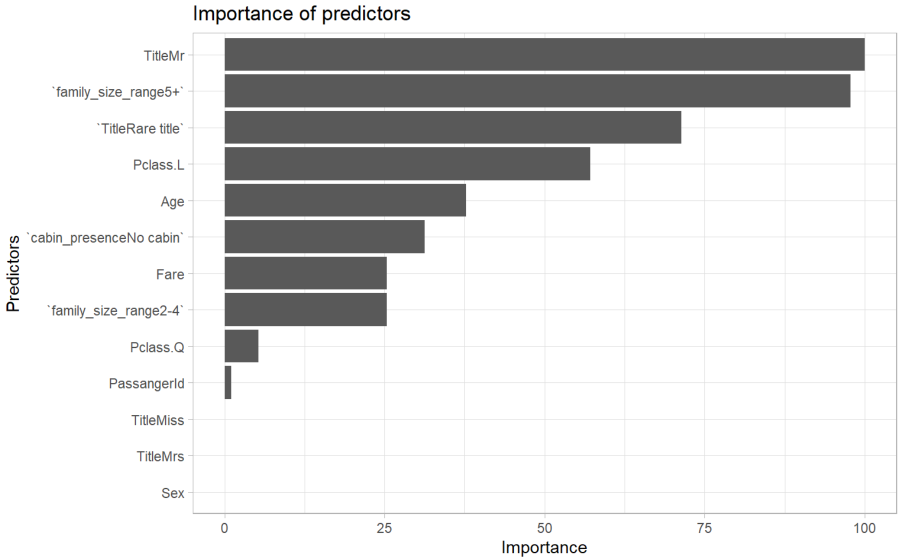
```
confusionMatrix(lr_model)
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentual average cell counts across re
samples)
##
##           Reference
## Prediction    0    1
##          0 54.2  9.9
##          1  7.4 28.5
##
##  Accuracy (average) : 0.8272
```

Check the importance of each variable in the logistic regression model.

```
lr_imporatance <- varImp(lr_model)

ggplot(lr_imporatance, aes(x = reorder(variable, importa
         nce), y = 'Importance'))+
  geom_bar(stat = "identity")+
  labs(title = "Importance of predictors",x = "Predictor
         s", y = "Importance")+
  theme_light()
```



Prediction: Logistic Regression Model.

Predicting using the test set.

```
predictio_lr <- predict(lr_model, test)
```

Save the solution to a data frame with two columns : PassengerId and Survived(Prediction)

```
solution_lr <- data.frame(PassangerID = test$PassangerI
         d, Survived = predictio_lr)
```

Write the solution to a file.

```
write.csv(solution_lr, file = "lr_Titanic_Solution.csv",
         row.names = FALSE)
```

> **Predicted accuracy of logistic regression model: 82.83%, Leader board accuracy: 63.288%.**

Results

Table of results:

Classifier	Predicted Accuracy	Leader board Accuracy
Random Forest	0.82899	0.79904
Logistic Regression	0.82714	0.77990

As we can see from the table, the **random forest** model showed the greatest accuracy on the leader board prediction.