Titanic - Machine Learning Kaggle Competition

Abdellah AitElmouden | Gabriel Abreu | Jered Ataky | Patrick Maloney

5/11/2021

Abstract

The purpose of our final project is to enter Kaggle's "Titanic - Machine Learning from Disaster" competition. The goal is to predict as accurately as possible which passengers aboard the Titanic survived the shipwreck. We chose this challenge as our final project because it is the culmination of all the skills and methodologies we learned during this semester. This project will use several techniques for the classification of each passenger as someone who either died or survived, including logistic regression and random forest modeling. A logistic regression model ended up producing the best results, outperforming or random forest models. Our best model received a score of 0.77511

Background and Challenge Description

"The sinking of the Titanic is one of the most infamous shipwrecks in history.

On April 15, 1912, during her maiden voyage, the widely considered "unsinkable" RMS Titanic sank after colliding with an iceberg. Unfortunately, there weren't enough lifeboats for everyone onboard, resulting in the death of 1502 out of 2224 passengers and crew.

While there was some element of luck involved in surviving, it seems some groups of people were more likely to survive than others.

In this challenge, we ask you to build a predictive model that answers the question: "what sorts of people were more likely to survive?" using passenger data (i.e. name, age, gender, socio-economic class, etc.)."(https://www.kaggle.com/c/titanic)

Literature Review

Below is a list of articles that helped us in determining our approach to this competition.

https://towardsdatascience.com/kaggles-titanic-competition-in-10-minutes-part-i-e6d18e59dbce

https://medium.datadriveninvestor.com/start-with-kaggle-a-comprehensive-guide-to-solve-the-titanic-challenge-8ac5815b0473

https://python.plainenglish.io/what-happened-when-i-used-stacking-on-the-kaggle-titanic-competition-7914b1b02d6c

https://datatricks.co.uk/80-in-kaggles-titanic-competition-in-50-lines-of-r-code https://towardsdatascience.com/predicting-titanic-survivors-a-kaggle-competition-625405f5031e

```
These articles provide a good overview of the current state-of-the-art for solving the Titanic Kaggle c
The literature also suggests that feature engineering is also a key to solving this problem. Most of th
```

Methodology

`Our methodology consisted of a 5-step process. The data was provided by Kaggle and was pre-separated in`

- Data Exploration: summary statistics and simple visualizations were created to search for relationships between the variables.
- Data Preparation: null values were imputed and new features were engineered.
- Logistic Regression Modeling: A binomial logistic regression model was used as an initial comparison for the following model that was simplified using stepwise regression.
- Random Forest Modeling: two different random forest functions were used from different packages to be sure we had the best version.
- Evaluation: the test data was then cleaned and run through the models and their performance was evaluated. Additional tweaks to the models were made in attempt to improve performance.

Experimentation and Results

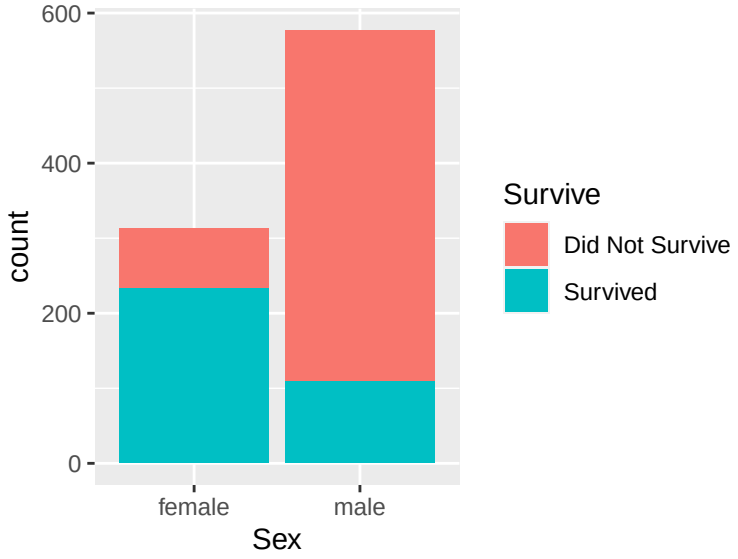
Data Exploration

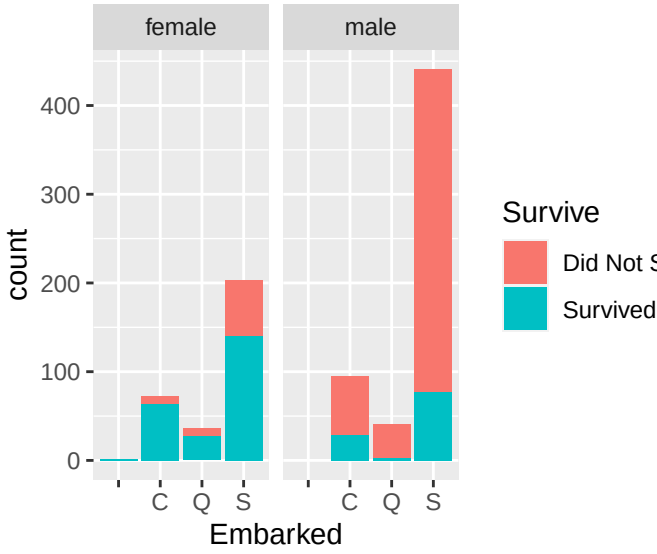
First, we loaded the data and examined the structure. The data has 5 factor, 5 discrete, and 2 continuous variables. There will need to be further exploration to see if any of the columns are missing data.

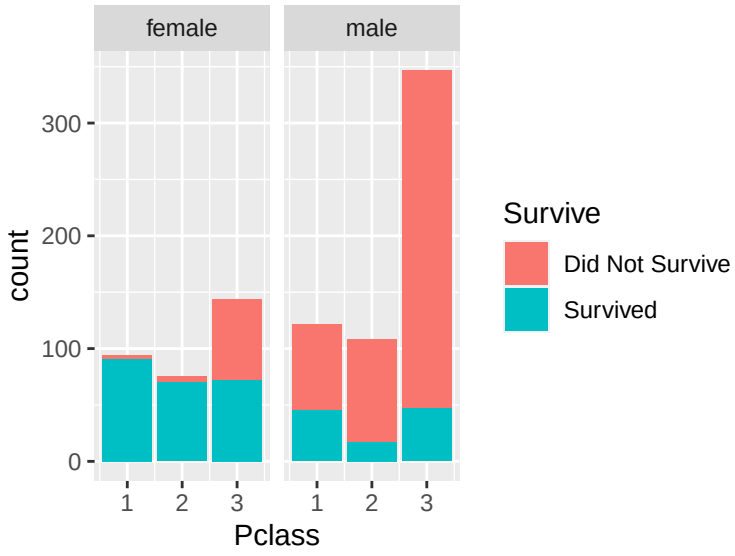
The Age column is only 80% complete with 177 missing values. We're going to have impute the missing data.

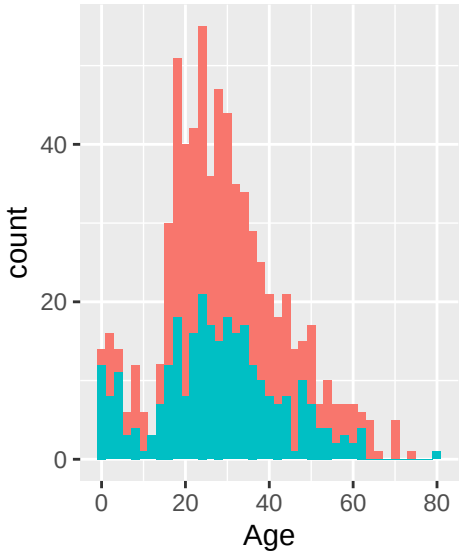
The Cabin column has 687 empty values, imputing that data may not be the best choice since we're filling in more than 50% of the empty values. On the other hand Embarked only has 2 empty values, so we can fill in those empty values without major impact.

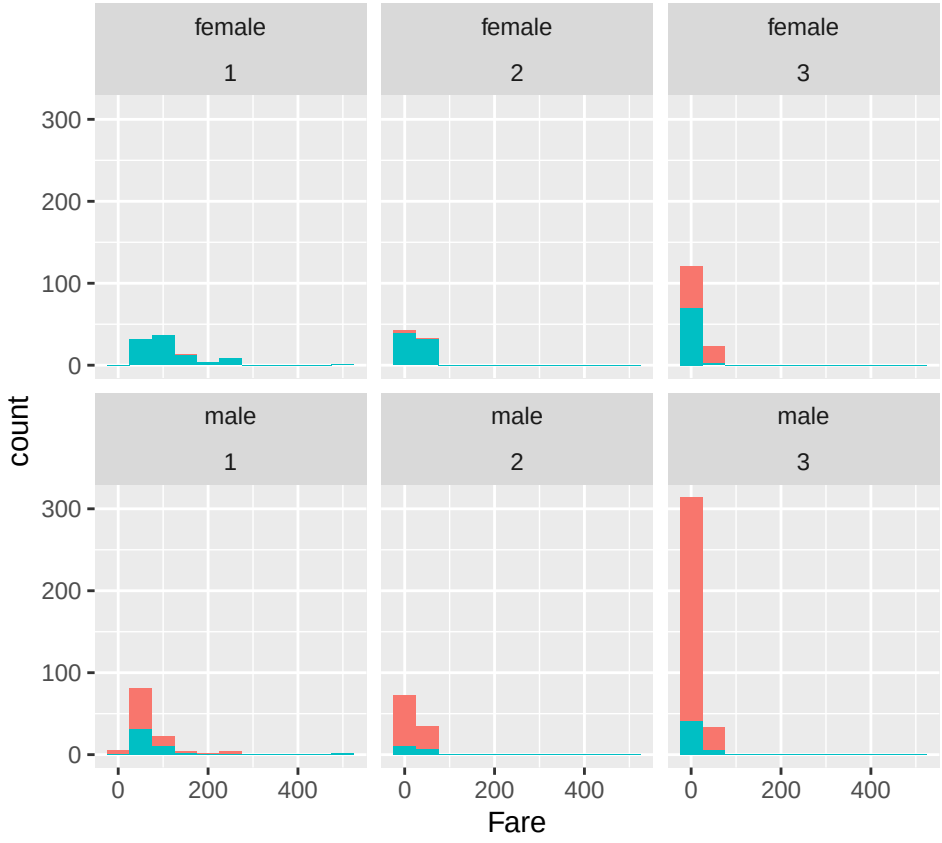
Visualization Below are several visualizations to help us see how survivors are distributed amongst certain variables.



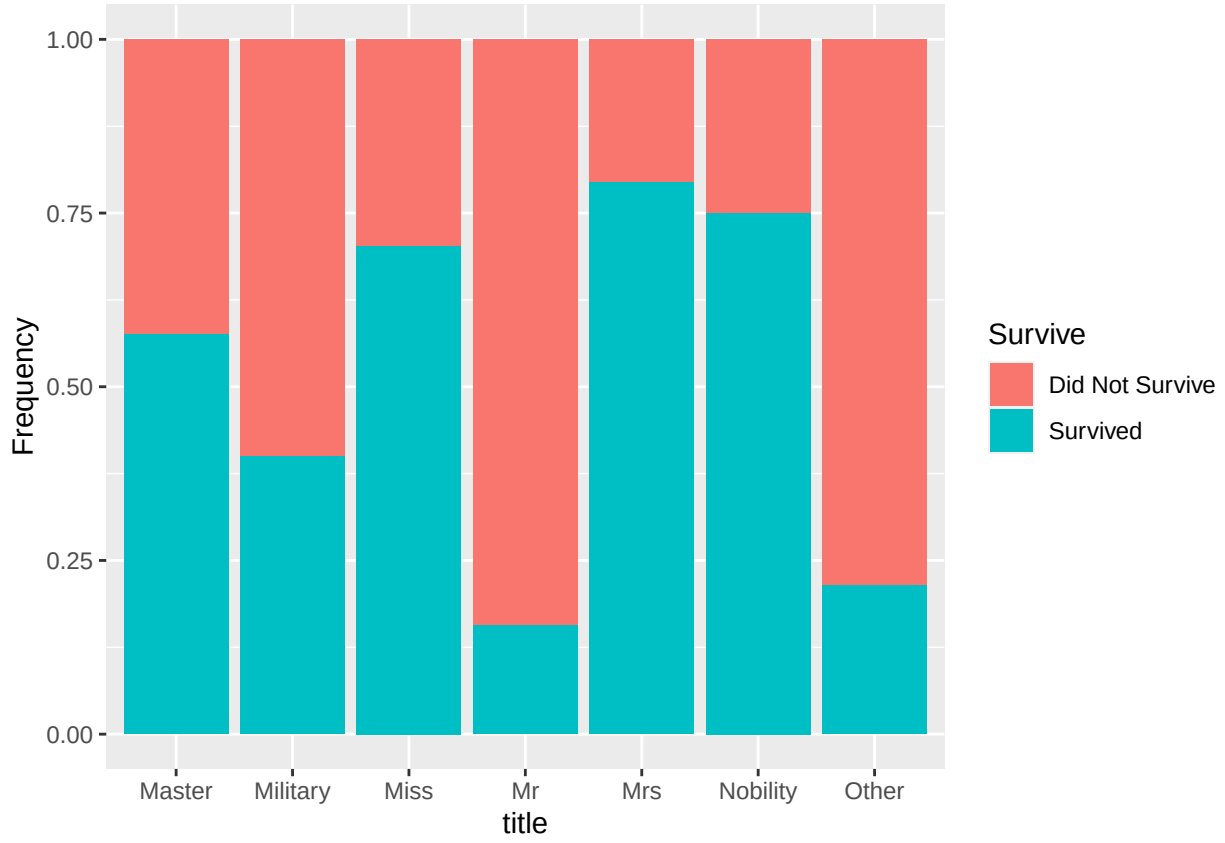






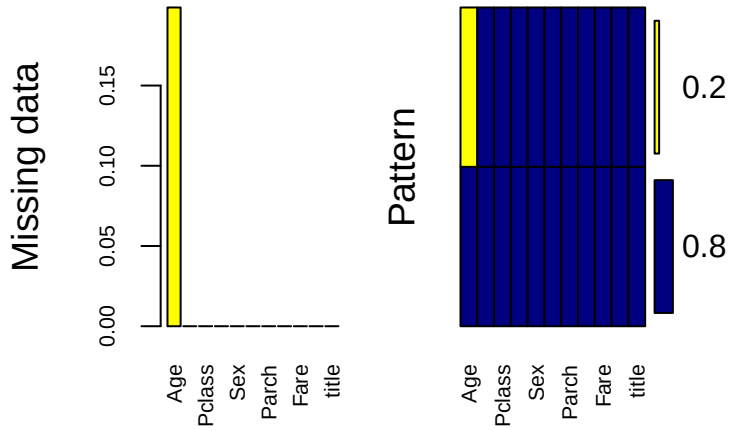


Based on the visuals, it seemed gender and class had an effect on a passenger's probability of surviving. We then looked at the titles associated with the passenger's name.



Data Preparation

There are three variables with missing or empty values based on our exploration of the data and visualizations: Embark, Cabin, and Age. Only passengers 62 and 830 are missing their embark ports. We randomly assigned them a value of "C". The column Cabin had too many missing values to impute or fill, so we dropped the Cabin column from the training data set.



```
##
##  Variables sorted by number of missings:
##  Variable     Count
##       Age 0.1986532
##  Survived 0.0000000
##    Pclass 0.0000000
##      Name 0.0000000
##       Sex 0.0000000
##     SibSp 0.0000000
##     Parch 0.0000000
##    Ticket 0.0000000
##      Fare 0.0000000
##  Embarked 0.0000000
##     title 0.0000000
```

We can see that Age was missing nearly 20% of its values. This data was imputed using the rpart function.

Models

Model 1: Binomial with logit link function (w/ Imputed data)

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + title + Embarked + Fare +
##     Age + SibSp + Parch, family = binomial(link = "logit"), data = train3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4068  -0.5433  -0.3760   0.5370   2.5761
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)    20.721916 481.641207   0.043  0.96568
## Pclass         -1.140275   0.161793  -7.048 1.82e-12 ***
## Sexmale       -15.092293 481.640636  -0.031  0.97500
## titleMilitary  -2.779350   1.133695  -2.452  0.01422 *
## titleMiss     -15.629763 481.640884  -0.032  0.97411
## titleMr        -3.405782   0.548291  -6.212 5.24e-10 ***
## titleMrs      -14.742217 481.640937  -0.031  0.97558
## titleNobility  -2.728641   1.529793  -1.784  0.07448 .
## titleOther     -3.995306   0.979950  -4.077 4.56e-05 ***
## EmbarkedQ      -0.103967   0.397173  -0.262  0.79350
## EmbarkedS      -0.414950   0.248222  -1.672  0.09459 .
## Fare            0.003291   0.002608   1.262  0.20689
## Age            -0.031319   0.009698  -3.229  0.00124 **
## SibSp          -0.569119   0.127482  -4.464 8.03e-06 ***
## Parch          -0.365867   0.136334  -2.684  0.00728 **
## ---
## 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
```

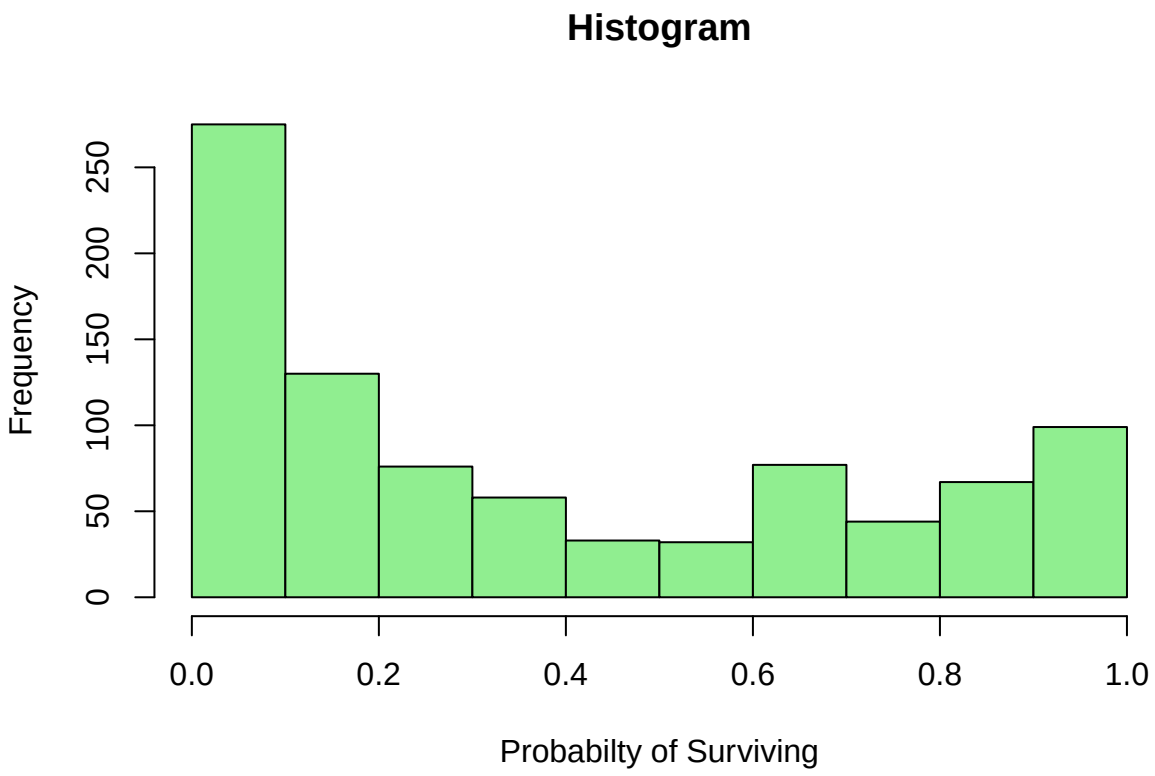
```
## Residual deviance:  722.12  on 876  degrees of freedom
## AIC: 752.12
##
## Number of Fisher Scoring iterations: 13
```

Model 2: Stepwise

```
## Start:  AIC=752.12
## Survived ~ Pclass + Sex + title + Embarked + Fare + Age + SibSp +
##     Parch
##
##            Df Deviance    AIC
## - Embarked  2   725.41 751.41
## - Fare      1   723.92 751.92
## <none>          722.12 752.12
## - Sex       1   726.54 754.54
## - Parch     1   729.88 757.88
## - Age       1   733.03 761.03
## - SibSp     1   747.11 775.11
## - title     6   780.25 798.25
## - Pclass    1   772.73 800.73
##
## Step:  AIC=751.41
## Survived ~ Pclass + Sex + title + Fare + Age + SibSp + Parch
##
##          Df Deviance    AIC
## <none>       725.41 751.41
## - Fare    1   728.49 752.49
## - Sex     1   729.60 753.60
## - Parch   1   733.84 757.84
## - Age     1   736.90 760.90
## - SibSp   1   753.19 777.19
## - title   6   783.55 797.55
## - Pclass  1   777.98 801.98

##
## Call:
## glm(formula = Survived ~ Pclass + Sex + title + Fare + Age +
##     SibSp + Parch, family = binomial(link = "logit"), data = train3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4747  -0.5493  -0.3854   0.5225   2.6670
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)    20.321420 481.372829   0.042 0.966327
## Pclass         -1.132427   0.157823  -7.175 7.21e-13 ***
## Sexmale       -14.986038 481.372309  -0.031 0.975164
## titleMilitary  -2.813948   1.128833  -2.493 0.012674 *
## titleMiss     -15.516876 481.372559  -0.032 0.974285
## titleMr        -3.444979   0.545960  -6.310 2.79e-10 ***
## titleMrs      -14.666556 481.372612  -0.030 0.975694
```

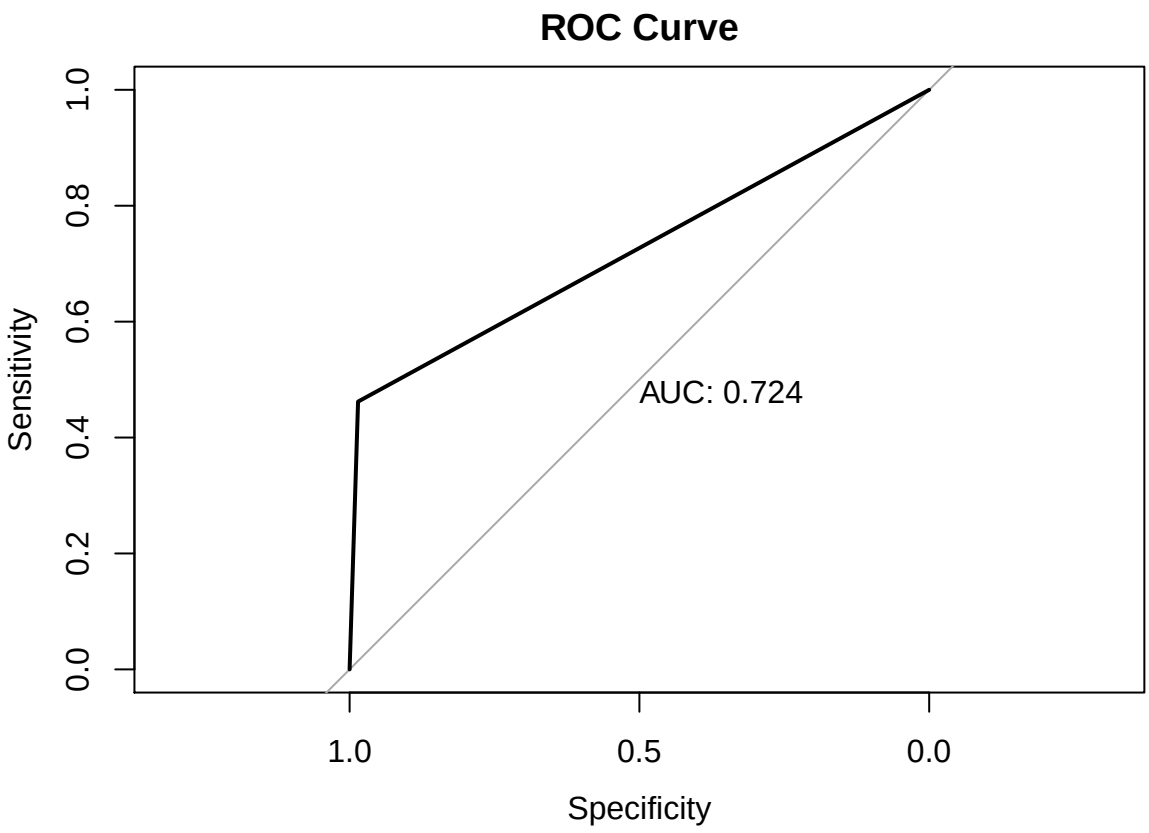
```
## titleNobility  -2.640764   1.543909  -1.710 0.087185 .
## titleOther     -3.920256   0.965141  -4.062 4.87e-05 ***
## Fare            0.004198   0.002594   1.618 0.105608
## Age            -0.031856   0.009633  -3.307 0.000943 ***
## SibSp          -0.591944   0.126686  -4.673 2.97e-06 ***
## Parch          -0.378039   0.135425  -2.791 0.005247 **
## ---
## 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:  725.41  on 878  degrees of freedom
## AIC: 751.41
##
## Number of Fisher Scoring iterations: 13
```




```
##      res.deviance  df         p
## [1,]     725.4096 878 0.9999434
```

```
##            Actual
## Predictions   0   1
##           0 541 184
##           1   8 158
```

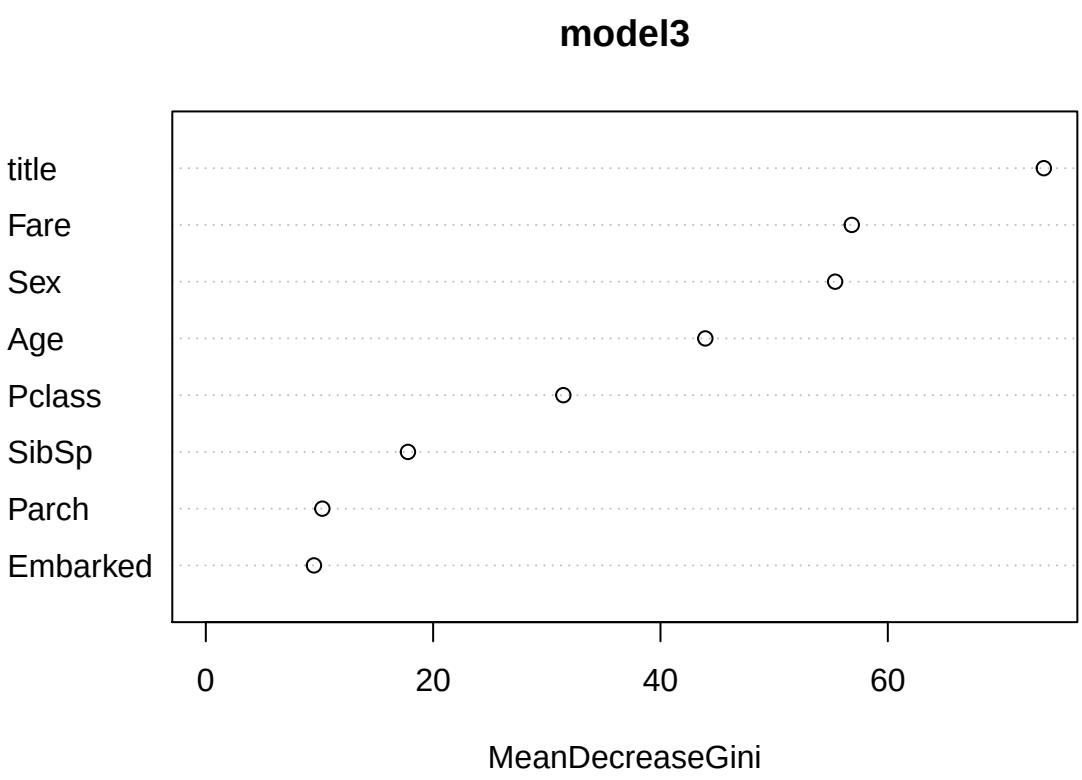
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 541 184
##          1   8 158
##
##                Accuracy : 0.7845
##                  95% CI : (0.756, 0.8111)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4955
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4620
##             Specificity : 0.9854
##          Pos Pred Value : 0.9518
##          Neg Pred Value : 0.7462
##              Prevalence : 0.3838
##          Detection Rate : 0.1773
##    Detection Prevalence : 0.1863
##       Balanced Accuracy : 0.7237
##
##        'Positive' Class : 1
##
```

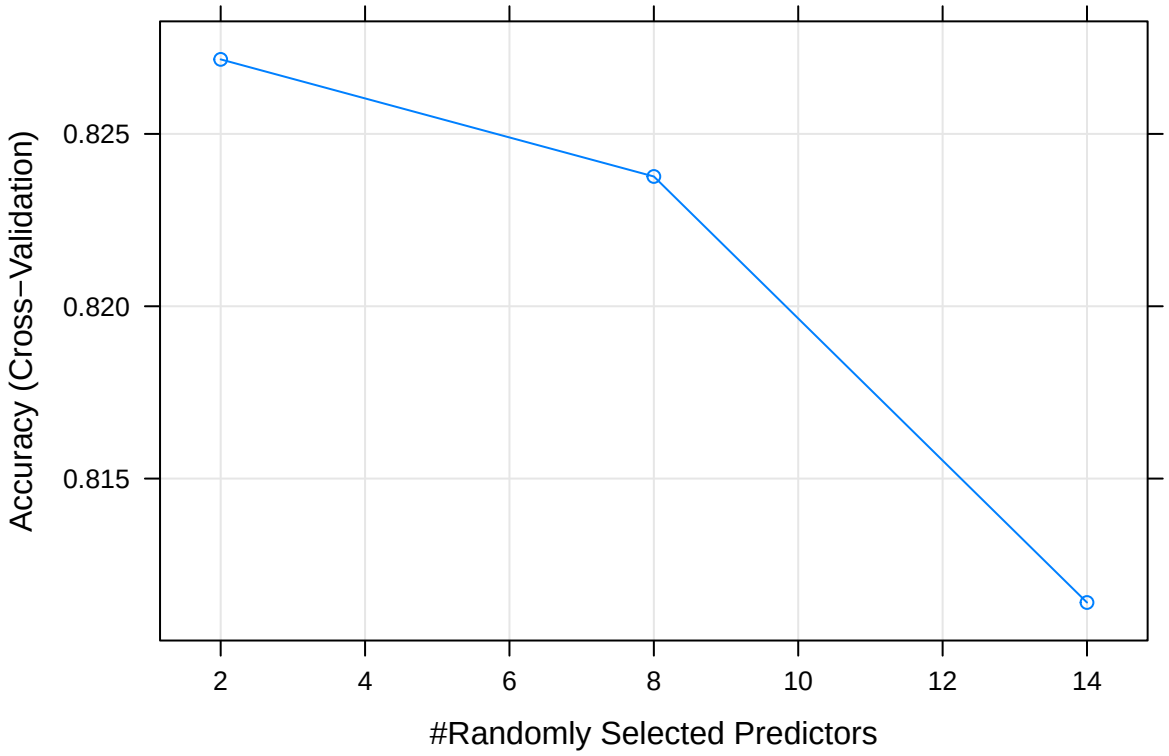


Model 3: Random Forest

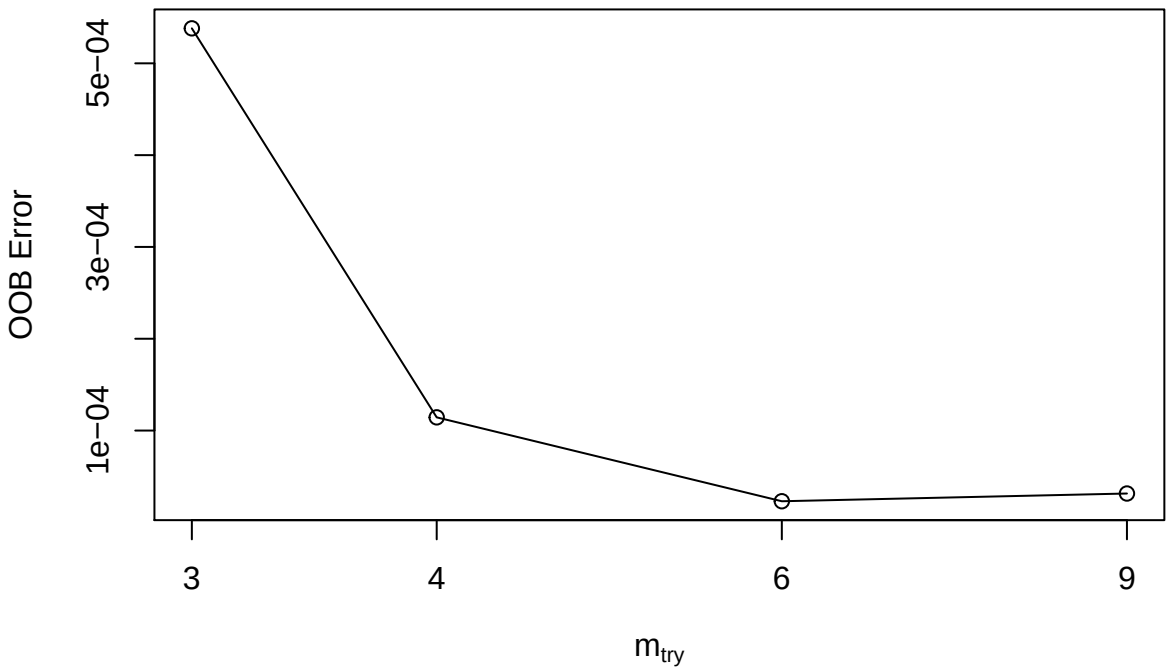
```
## Random Forest
##
## 891 samples
##   8 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 802, 801, 802, 802, 801, 803, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.8271624  0.6270858
##    8    0.8237652  0.6230913
##   14    0.8114053  0.5968540
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.

## integer(0)
```


model3
title
Fare
Sex
Age
Pclass
SibSp
Parch
Embarked
0
20
40
60
MeanDecreaseGini

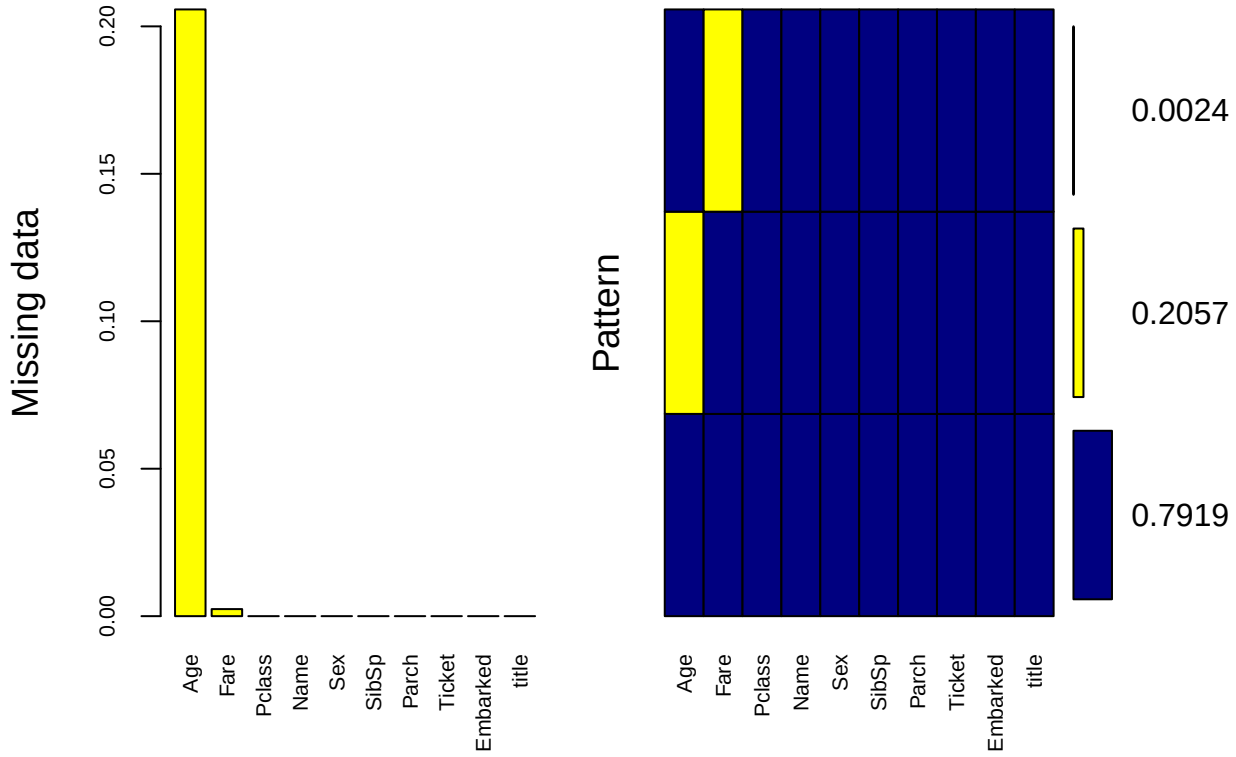



```
## -3.704311 0.01
## 0.7988898 0.01
## -0.3660311 0.01
```



Test Models

The same data cleaning steps were repeated for the test data as the training data.The null values were imputed and the title categories were created




```
##
##  Variables sorted by number of missings:
##  Variable       Count
##       Age 0.205741627
##      Fare 0.002392344
##    Pclass 0.000000000
##      Name 0.000000000
##       Sex 0.000000000
##     SibSp 0.000000000
##     Parch 0.000000000
##    Ticket 0.000000000
##  Embarked 0.000000000
##     title 0.000000000
```

Interestingly, Fare is missing values as well as Age in the test dataset. In the training data set, only Age had a significant amount of missing values. We imputed these values in Rpart as well as age.

Running the Models on the Test Set Second Model Predictions

Third Model Predictions

Random Forest with train method

Random Forest with randomForest function

Select Model

The model with the most accurate result is model 2. Tweaking the thresholds changed final results on kaggle but changing the threshold to 0.75 seemed to be optimal (produced score of 0.77511), while the random forest models produced scores of 0.75358.

The models might see greater accuracy testing different methods of imputation. The age column saw the greatest amount of missing values, focusing on creating accurate age values will most likely improve the models.

Appendix

```
library(corrplot)
library(tidyverse)
library(Hmisc)
library(PerformanceAnalytics)
library(mice)
library(gt)
library(caret)
library(bnstruct)
library(VIM)
library(corrr)
library(kableExtra)
library(rpart)
library(gtsummary)
library(reshape)
library(pROC)
library(randomForest)
library(pscl)
#Import the training and testing data sets:
train <- read.csv("https://raw.githubusercontent.com/aaitelmouden/DATA621/master/Final%20Project/train.
test <- read.csv("https://raw.githubusercontent.com/aaitelmouden/DATA621/master/Final%20Project/test.cs
glimpse(train)
summary(train)
train_visual <- train%>%mutate(Survive = case_when(Survived==1~"Survived", Survived==0~"Did Not Survive
# Examine the survival rate for the overall population
prop.table(table(train_visual$Survive))
train_visual%>%ggplot(aes(Sex,fill=Survive))+geom_bar()
train_visual%>%ggplot(aes(Embarked,fill=Survive))+geom_bar()+facet_wrap(~Sex)
train_visual%>%ggplot(aes(Pclass, fill=Survive)) + geom_bar()+facet_wrap(~Sex)
train_visual%>%ggplot(aes(Age, fill=Survive)) + geom_histogram(binwidth = 2)
train_visual %>% ggplot(aes(Fare, fill=Survive))+geom_histogram(binwidth = 50)+facet_wrap(Sex~Pclass)
#Create a new column with all the different titles
train$title <- gsub('(.*, )|(\\..*)', '', train$Name)
table(train$Sex, train$title)
military_title <-c('Capt', 'Col', 'Major')
royal_title <-c('the Countess', 'Jonkheer', 'Sir', 'Lady')
the_rest <- c('Dr', 'Don', 'Rev')
the_master <- c('Master')
train$title[train$title=='Mlle']<-'Miss'
train$title[train$title=='Ms']<- 'Miss'
train$title[train$title=='Mme']<-'Mrs'
train$title[train$title %in% the_master] <- 'Master'
train$title[train$title %in% military_title] <- 'Military'
```

```
train$title[train$title %in% royal_title] <- 'Nobility'
train$title[train$title %in% the_rest] <- 'Other'
table(train$Sex, train$title)
train$title <- as.factor(train$title)
train_visual$title <- train$title
train_visual%>%ggplot(aes(title, fill=Survive)) + geom_bar(position = 'fill') + ylab('Frequency')
#Localization of the empty values for Embark
train$Embarked[train$Embarked == ""] <- NA
train[(which(is.na(train$Embarked))), 1]
# Only passengers 62 and 830 are missing their embark ports. We will randomly assign them "C".
train$Embarked[c(62, 830)] <- 'C'
train2 <- subset(train, select = -c(Cabin, PassengerId))
aggr(train2, col=c('navyblue','yellow'),
numbers=TRUE, sortVars=TRUE,
labels=names(train2), cex.axis=.7,
gap=3, ylab=c("Missing data","Pattern"))
#We can impute the age date using the rpart function.
#source: https://cran.r-project.org/web/packages/rpart/vignettes/longintro.pdf
train3 <- train2
predicted_age <- rpart(Age ~ Pclass + Sex + SibSp + Parch + Fare + Embarked + title,
                       data = train3[!is.na(train3$Age),], method = "anova")
train3$Age[is.na(train3$Age)] <- predict(predicted_age, train3[is.na(train3$Age),])

#### Model 1: Binomial with logit link function (w/ Imputed data)
model1 <- glm(Survived ~ Pclass + Sex + title + Embarked + Fare + Age + SibSp + Parch, family = binomial
summary(model1)
## Model 2
model2 <- step(model1)
summary(model2)
hist(model2$fitted.values, main="Histogram", xlab="Probabilty of Surviving", col="light green")
with(model2, cbind(res.deviance = deviance, df = df.residual,  p = pchisq(deviance, df.residual, lower.t
train4 <- train3
probabilities <- predict(model2, train4, type = "response")
predicted.classes <- ifelse(probabilities > 0.8, 1, 0)
train4$pred.class <- predicted.classes
table("Predictions" = train4$pred.class, "Actual" = train4$Survived)
confusionMatrix(as.factor(predicted.classes), as.factor(train4$Survived), positive = '1')
curve <- roc(response = train4$Survived,
    predictor = predicted.classes,
    plot = TRUE,
    print.auc = TRUE,
    main = "ROC Curve")
#### Model 3: Random Forest
set.seed(51)
#model3 <- randomForest(factor(Survived) ~ Pclass + title + Sex + Fare + SibSp + Parch + Age + Embarked
model3.1 <- train(factor(Survived) ~ Pclass + title + Sex + Fare + SibSp + Parch + Age + Embarked,
                  data = train3,
                  method = 'rf',
                  trControl = trainControl(method = 'cv',
                                           number = 10))
model3.1
which.min(model3$mse)
varImpPlot(model3)
plot(model3.1)
```

```
model_tuned <- tuneRF(
               x=train3[,c(-3, -8)], #define predictor variables
               y=train3$Survived, #define response variable
               ntreeTry=500,
               mtryStart=4,
               stepFactor=1.5,
               improve=0.01,
               trace=FALSE #don't show real-time progress
               )
# Repeating the same data cleaning steps for the test data as the training data.
summary(test)
test$title <- gsub('(.*, )|(\\..*)', '', test$Name)
table(test$Sex, test$title)
military_title <-c('Capt', 'Col', 'Major')
royal_title <-c('the Countess', 'Jonkheer', 'Sir', 'Lady')
the_rest <- c('Dr', 'Don', 'Rev', 'Dona')
the_master <- c('Master')
test$title[test$title=='Mlle']<-'Miss'
test$title[test$title=='Ms']<- 'Miss'
test$title[test$title=='Mme']<-'Mrs'
test$title[test$title %in% the_master] <- 'Master'
test$title[test$title %in% military_title] <- 'Military'
test$title[test$title %in% royal_title] <- 'Nobility'
test$title[test$title %in% the_rest] <- 'Other'
table(test$Sex, test$title)
test$title <- as.factor(test$title)
#Checking for Embarked missing passengers, which there are none.
test$Embarked[test$Embarked == ""] <- NA
test[(which(is.na(test$Embarked))), 1]
test2 <- subset(test, select = -c(Cabin, PassengerId))
aggr(test2, col=c('navyblue','yellow'),
numbers=TRUE, sortVars=TRUE,
labels=names(test2), cex.axis=.7,
gap=3, ylab=c("Missing data","Pattern"))
test3 <- test2
predicted_age_test <- rpart(Age ~ Pclass + Sex + SibSp + Parch + Fare + Embarked + title,
                      data = test3[!is.na(test3$Age),], method = "anova")
test3$Age[is.na(test3$Age)] <- predict(predicted_age_test, test3[is.na(test3$Age),])
predicted_fare_test <- rpart(Fare ~ Pclass + Sex + SibSp + Parch + Age + Embarked + title,
                      data = test3[!is.na(test3$Fare),], method = "anova")
test3$Fare[is.na(test3$Fare)] <- predict(predicted_fare_test, test3[is.na(test3$Fare),])
#Second Model Predictions
test4 <- test3
pred.test2 <- predict(model2, test4, type = "response")
predtest.classes <- ifelse(pred.test2 > 0.75, 1, 0)
test4$pred.class <- predtest.classes
sub1 <- data.frame(test$PassengerId, test4$pred.class)
colnames(sub1) = c("PassengerId", "Survived")
#Third Model Predictions
#source: https://stackoverflow.com/questions/24829674/r-random-forest-error-type-of-predictors-in-new-da
xtest <- rbind(train3[1,-1], test3)
xtest <- xtest[-1,]
#Random Forest with train method
pred.test3.1 <- predict(model3.1, newdata=xtest, type="raw")
```

```
sub2 <- data.frame(test$PassengerId, pred.test3.1)
colnames(sub2) = c("PassengerId", "Survived")
#Random Forest with randomForest function
pred.test3 <- predict(model3, newdata = xtest, type = "response")
sub3 <- data.frame(test$PassengerId, pred.test3)
colnames(sub3) = c("PassengerId", "Survived")
```