Titanic Disaster

Robert Sneiderman

February 2020

This tutorial will teach you how to obtain an accuracy of 0.79425 on Kaggle's "Titanic Data Set". We will due this using R. For all the code, please check out my GitHub!

As of March 3, 2020, this score is in the top 12%. Note that the actual testing data is available so you could obtain an accuracy of 1 by simply plugging them in (but that would defeat the point of the project).

At the bottom of the document is extra information that you may find relevant but that were not used in the final prediction.

The Titanic Dataset consists 891 observations and 14 columns. The goal is to predict if a passenger survived given their unique attributes. Some of the attributes are; Sex, Age and Class.

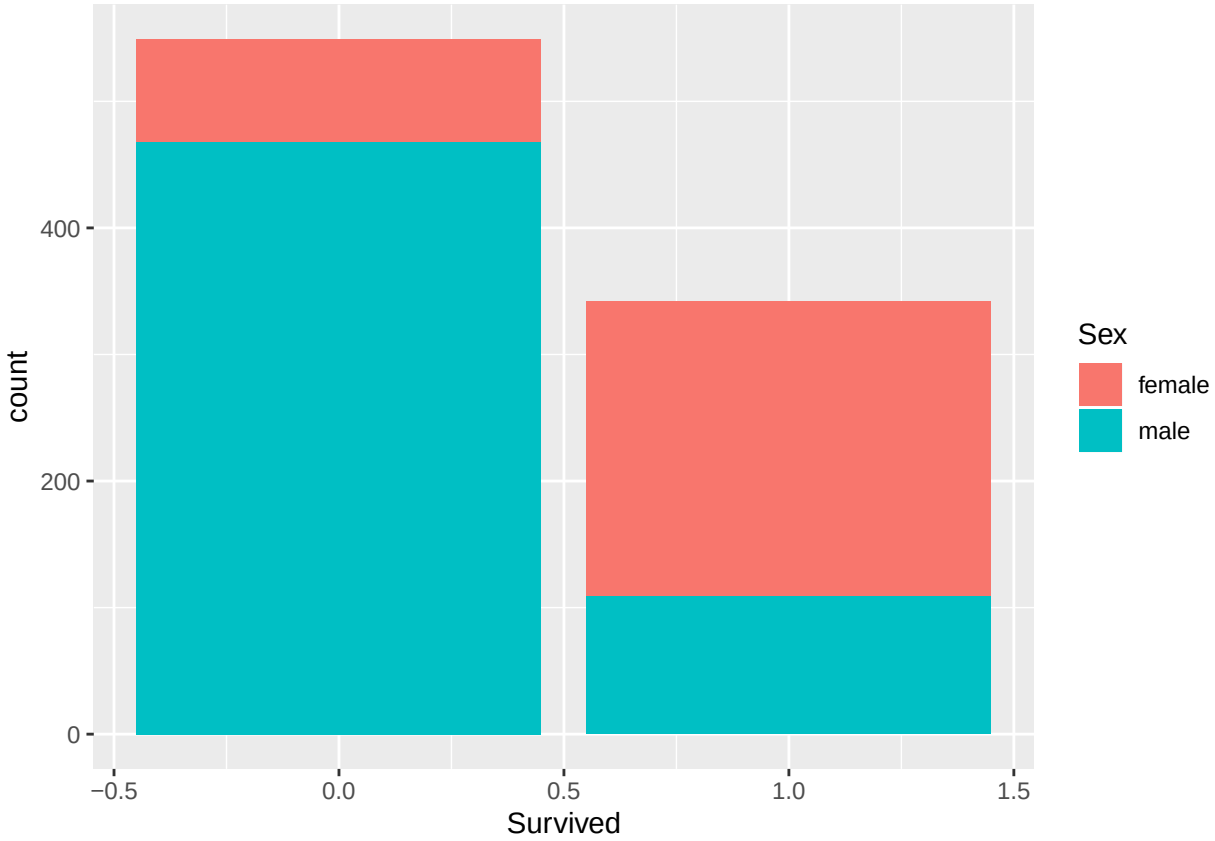
Firstly, Load packages and Read in Data

```
suppressMessages(library(tidyverse))
suppressMessages(library(caret))
suppressMessages(library(ggplot2))

#Make sure to set the correct  working directory where your files are saved in a csv format.

titanic=read.csv("titanic_train.csv")
titanic_testing=read.csv("titanic_test.csv")



titanic_testing$Survived=NA #This is what we want to predict eventually
titanic_testing=titanic_testing[ ,c(12,1:11)] #Simple reorder of the columns

titanic_full=rbind(titanic,titanic_testing)
# Create a full data set to use for imputing missing values and or feature engineering.
```

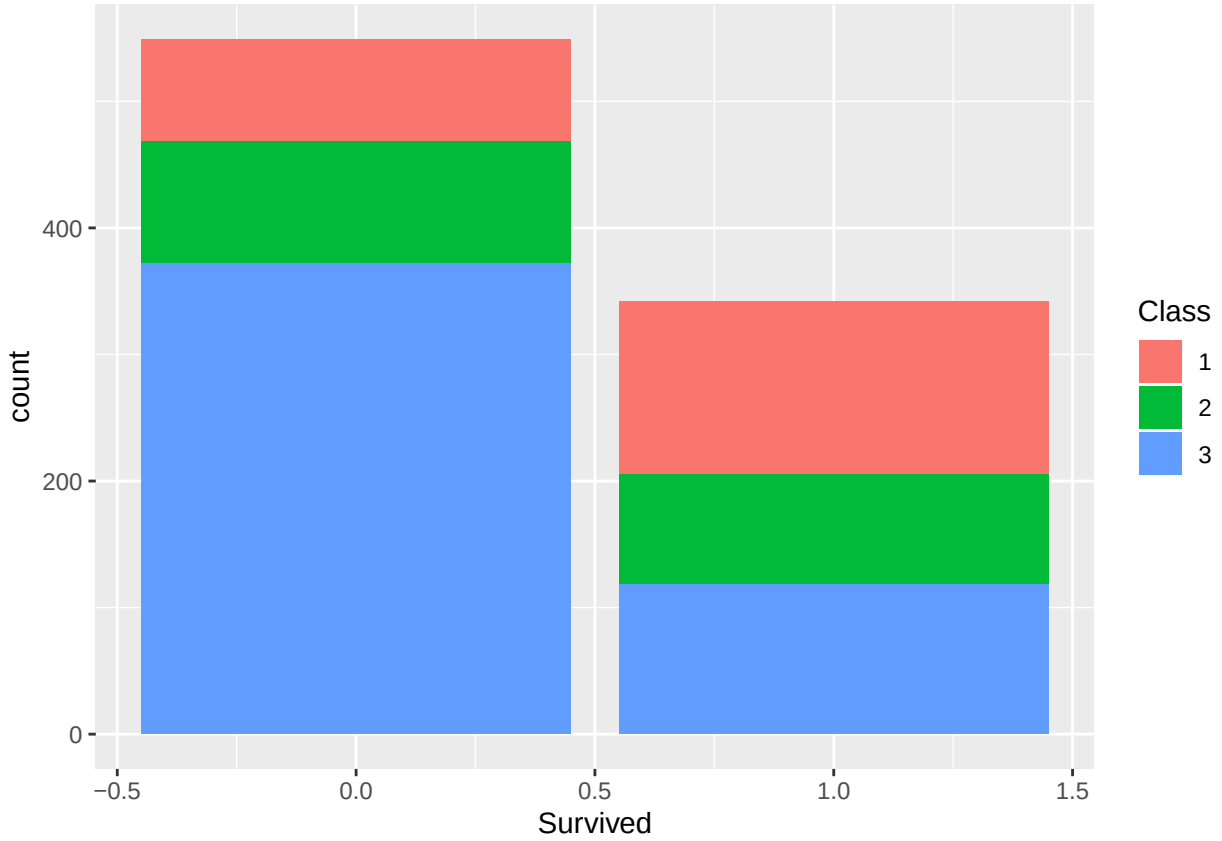
Let us take a quick look at the training dataset to see what we are working with. The majority of the training data consists of men, although there are many women as well.

```
## [1] "There are: 577 Men"
```

```
## [1] "There are: 314 Females"
```



Among the deceased is mostly men and the majority of survivors were women. “Sex” will be an important predictor.



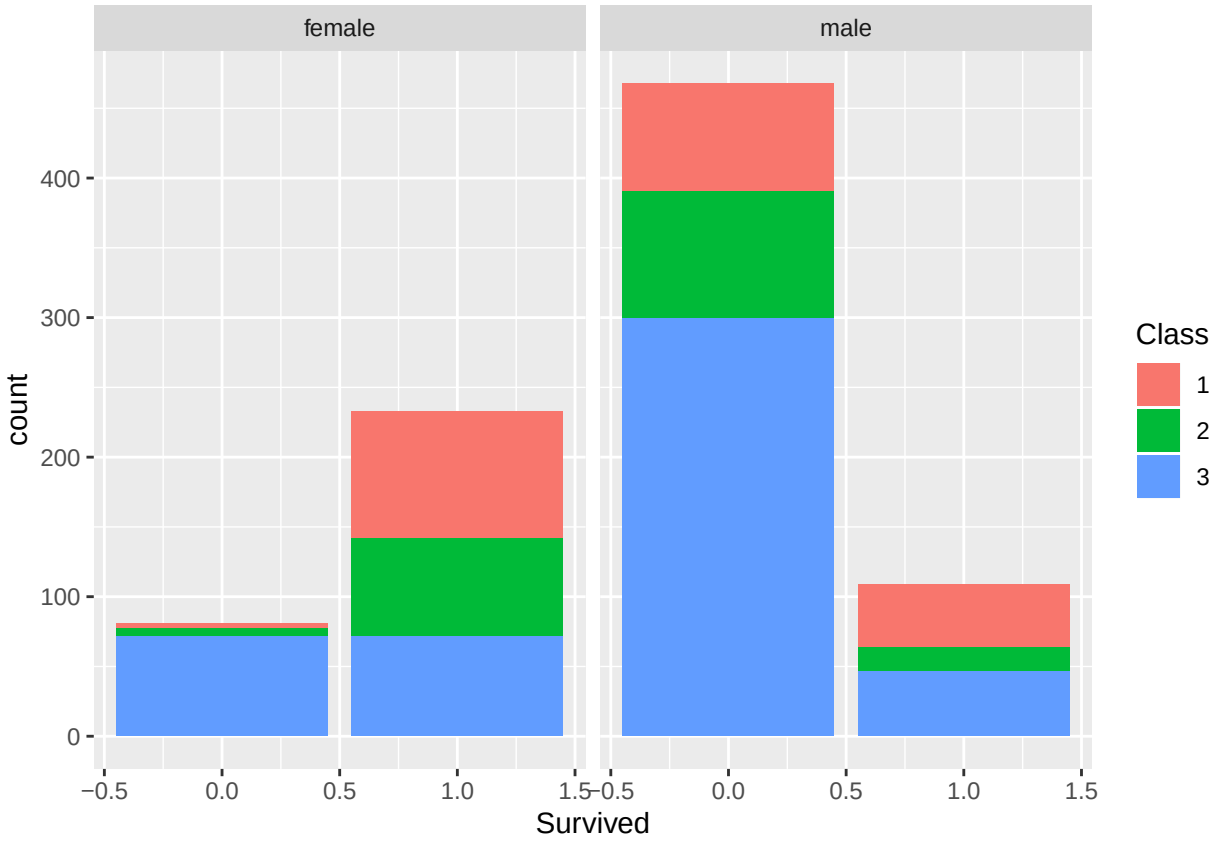

```
## [1] "There are: 216 First class passengers"
```

```
## [1] "There are: 184 Second class passengers"
```

```
## [1] "There are: 491 Third class passengers"
```

Even though first class had the least amount of passengers, they are way highly represented among the surviors. Class is an important predictor!

Let us also take a look at class WITH sex.



Notice that almost no first class women died!

Only one value of "Fare" is missing, we can simply impute the value based off the mean of Fare for this persons class (third class).

```
titanic_full[1044,'Fare']=mean(titanic_full[titanic_full$Pclass==3,]$Fare,na.rm=T)
#Impute the one missing value of Fare based on their class.
titanic_testing[153,'Fare']=mean(titanic_full[titanic_full$Pclass==3,]$Fare,na.rm=T)
```

Many values of "Age" are missing. We will try these missing values as a factor and create a new column AgeF to represent a range of ages. Let us turn age into a broad factor as well,we will consider NA a possible category that may convey some information.The alternative to this method would be to try and predict the missing age value using other predictors.

```
titanic_full$AgeF=cut(titanic_full$Age,seq(0,max(titanic_full$Age,na.rm=T),5))
#We will consider age as a factor split, the groups will be
#0-5,5-10,. Plus a NA category that will represent missing age.


titanic_full$AgeF=addNA(titanic_full$AgeF)
# Important! Need to add NA as an actual factor level.


titanic$AgeF=titanic_full$AgeF[1:nrow(titanic)]
```

```
titanic_testing$AgeF=titanic_full$AgeF[892:1309]
```

We engineer a new feature, "Title" based off information in the "Name" column. Since several "titles" appear and clearly convey more information than simply a name can at this level of analysis.

Let us make a new column, a factor variable denoting the title of the passenger.Titles include: Mr, Mrs, Miss, Master,Rev,Dr,Lady, Countess, Dona,Capt, Jonkheer, Majorand Col. These are obtained from the "Name" column and likely will convey more information than simply a name.

```
#Make new column, Title
titanic_full$Title=NA


titanic_full$Title <- gsub('(.*, )|(\\..*)','',titanic_full$Name)
#This trick is from Huijun Zhao, leaves everything but the skeleton of the titles.
#The reason this works is because all possible titles are sorrounded by either
#of the two above regular expressions.
#It could also be done manually using grep.



titanic_full$Title[titanic_full$Title %in% c('Miss','Ms','Mlle')]<-"Miss"
titanic_full$Title[titanic_full$Title %in% c("Capt","Dr","Rev","Jonkheer","Major","Col","Sir","
  "Special men"
titanic_full$Title[titanic_full$Title %in% c('Master')]<-'Master'
titanic_full$Title[titanic_full$Title %in% c("Mr")]<-"Mr"
titanic_full$Title[titanic_full$Title %in% c("Dona","Lady","the Countess")]<-"Special women"
titanic_full$Title[titanic_full$Title %in% c("Mrs","Mme")]<-"Mrs"



#Turn Title into factor

titanic_full$Title=as.factor(titanic_full$Title)

titanic$Title=titanic_full$Title[1:nrow(titanic)]

titanic_testing$Title=titanic_full$Title[892:1309]
```

Modeling:

```
titanic$Survived=as.factor(titanic$Survived) #Make sure survived is a factor not a numeric.


titanic.n=titanic %>% #For modeling we remove columns which don't seem useful.
select(-c(PassengerId,Name,Ticket,Cabin,Age,Embarked))
#Age we have as a factor, we used title not name, etc.
```

```r
fitControl <- trainControl(method = "cv", #We will do 5 cross validation to improve accuracy
                           number = 5)



model_titanic_logistic=caret::train(Survived~.,data=titanic.n,method='glm',
                                    family='binomial',trControl=fitControl)



logistic_pred=predict(model_titanic_logistic,titanic_testing)


#Prepare to submit for kaggle
pid=titanic_testing$PassengerId # Get a vector of passenger Ids for the testing set
df.sub=data.frame(pid,logistic_pred)
names(df.sub)=c("PassengerId","Survived")
```

Logistic Regression gave an Accuracy of 0.79425 once we submit to Kaggle! Pretty good. In the end, logistic regression provided better predictions compared to Gbm, KNN or random forests.

Extra Material

Cabin:

Let Us consider cabin now, the possible cabins are 1st class had the top decks (A-E), second class (D-F), and third class (E-G). It also makes sense that the people towards the top (higher decks, higher Pclass) likely survived. If blank, turn into a factor in the same way we did for Age.

```
#ggM=grep("",titanic_full$Cabin) #These will represent a missing Cabin.

#ggA=grep("A",titanic_full$Cabin)
#ggB=grep("B",titanic_full$Cabin)
#ggC=grep("C",titanic_full$Cabin)
#ggD=grep("D",titanic_full$Cabin)
##ggE=grep("E",titanic_full$Cabin)
#ggF=grep("F",titanic_full$Cabin)
#ggG=grep("G",titanic_full$Cabin)
#ggT=grep("T",titanic_full$Cabin)
#titanic_full$CabinF=NA


#titanic_full$CabinF[ggM]="M"
#titanic_full$CabinF[ggA]="A"
#titanic_full$CabinF[ggB]="B"
#titanic_full$CabinF[ggC]="C"
#titanic_full$CabinF[ggD]="D"
#titanic_full$CabinF[ggE]="E"
#titanic_full$CabinF[ggF]="F"
#titanic_full$CabinF[ggG]="G"
#titanic_full$CabinF[ggT]="T"

#titanic$CabinF=titanic_full$CabinF[1:891]
#titanic_testing$CabinF=titanic_full$CabinF[892:1309]
```

Family Size: Engineer a new variable, FamilySize that simply sumbs the number of siblings and parents for each passenger.

```
#titanic_full$FamilySize=NA
#for(i in 1:nrow(titanic_full)){
#titanic_full$FamilySize[i]=sum(titanic_full[i,'SibSp']+titanic_full[i,'Parch'])
#}


#titanic_full$FamilyF=as.factor(titanic_full$FamilySize)
#titanic$FamilyF=titanic_full$FamilyF[1:891]
#titanic_testing$FamilyF=titanic_full$FamilyF[892:1309]
```

Previously, we treated age as a factor and hence NA were a special category. Now, instead we attempt to model Age using other variables such as pclass, Siblings, etc. We will compare this with our previous estimate.

```
#titanic_removena=titanic_full%>% #Keep only non NA observations of Age
 # filter(!(is.na(titanic_full$Age)))

#titanic_na_age=titanic_full%>%
 #filter(is.na(titanic_full$Age))


#titanic_testing_na=titanic_testing%>%
 # filter((is.na(titanic_testing$Age)))

#model_age=caret::train(Age~Title+SibSp+Parch,method='rf',trControl=fitControl,
#data=titanic_removena,tuneLength=5) #Model age using random forest


#new_pred=predict(model_age,titanic_na_age)


#titanic_full$Age[which(is.na(titanic_full$Age))]=new_pred

#titanic=titanic_full[1:891, ]
#titanic_testing=titanic_full[892:1309, ]

#titanic$Survived=as.factor(titanic$Survived)

#model_surv=caret::train(Survived~Sex+Age+Pclass+SibSp+Fare+Title,method="glm",
#family="binomial",trControl=fitControl,data=titanic)



#pnew=predict(model_surv,titanic_testing)
```

Other Models

```
#Other models we can try but gave lower accuracy, K-Nearest Neighbour, Random Forset and Desic

#model_titanic3=caret::train(Survived~.,data=training,method='knn',trControl=fitControl)


#model_titanic4=caret::train(Survived~.,data=titanic,method='rf',trControl=fitControl)


#model_titanic5=caret::train(Survived~.,data=titanic,method='rpart',trControl=fitControl)
```