Titanic Disaster

Vasil Yordanov

26 June 2017

Logistic Regression without any feature engineering

This is my first Kaggle kernel. Here I am going to predict who is going to survive the Titanic distaster using logistic regression. First let's load the datasets provided by Kaggle.

```
train = read.csv("train.csv", stringsAsFactors = F)
test = read.csv("test.csv", stringsAsFactors = F)
```

Next let's inspect the training dataset, I are currently not focusing on the testing dataset as its structure is equivalent with the training dataset and only the number of observations will be different.

```
str(train)
```

```
## 'data.frame':    891 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)"
##  $ 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" ...
```

Here is the place to include some information about the variables:

- PassengerId - ID of the passenger (this will definately be exluded from our regression model)
- Survived - factor variable indicating if the passenger survived (1) or not (0) the accident
- Pclass - factor variable indicating the ticket class 1st (upper), 2nd (middle) or 3rd (lower)
- Name - name of the passenger
- Sex - gender of the passenger
- Age - fractional if less than 1 and in form xx.5 if estimated
- SibSp - number of siblings / spouses aboard the Titanic
- Parch - number of parents / children aboard the Titanic
- Ticket - ticket number
- Fare - passenger fare
- Cabin - cabin number
- Embarked - facotr variable indicating the port of embarkation C (Cherbourg), Q (Queenstown), S (Southampton)

This reminds me that I need to convert some variables:

```
train$Survived = as.factor(as.character(train$Survived))
train$Pclass = as.factor(as.character(train$Pclass))
train$Sex = as.factor(as.character(train$Sex))
train$Embarked = as.factor(as.character(train$Embarked))
```

```
test$Pclass = as.factor(as.character(test$Pclass))
test$Sex = as.factor(as.character(test$Sex))
test$Embarked = as.factor(as.character(test$Embarked))
```

Let's continue with our data exploration:

```
summary(train)
```

```
##   PassengerId    Survived Pclass      Name               Sex
##  Min.   :  1.0   0:549    1:216   Length:891         female:314
##  1st Qu.:223.5   1:342    2:184   Class :character   male  :577
##  Median :446.0            3:491   Mode  :character
##  Mean   :446.0
##  3rd Qu.:668.5
##  Max.   :891.0
##
##       Age            SibSp           Parch           Ticket
##  Min.   : 0.42   Min.   :0.000   Min.   :0.0000   Length:891
##  1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000   Class :character
##  Median :28.00   Median :0.000   Median :0.0000   Mode  :character
##  Mean   :29.70   Mean   :0.523   Mean   :0.3816
##  3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##  Max.   :80.00   Max.   :8.000   Max.   :6.0000
##  NA's   :177
##       Fare           Cabin           Embarked
##  Min.   :  0.00   Length:891          :  2
##  1st Qu.:  7.91   Class :character   C:168
##  Median : 14.45   Mode  :character   Q: 77
##  Mean   : 32.20                      S:644
##  3rd Qu.: 31.00
##  Max.   :512.33
##
```

From the summary I can see that there are 177 missing observations for the Age variable. I need to investigate this further in order to decide what to do with these 177 observations. Thus I will make a subset of the training dataset containing only the observations with missing age.

```
missing = subset(train, is.na(Age))
summary(missing)
```

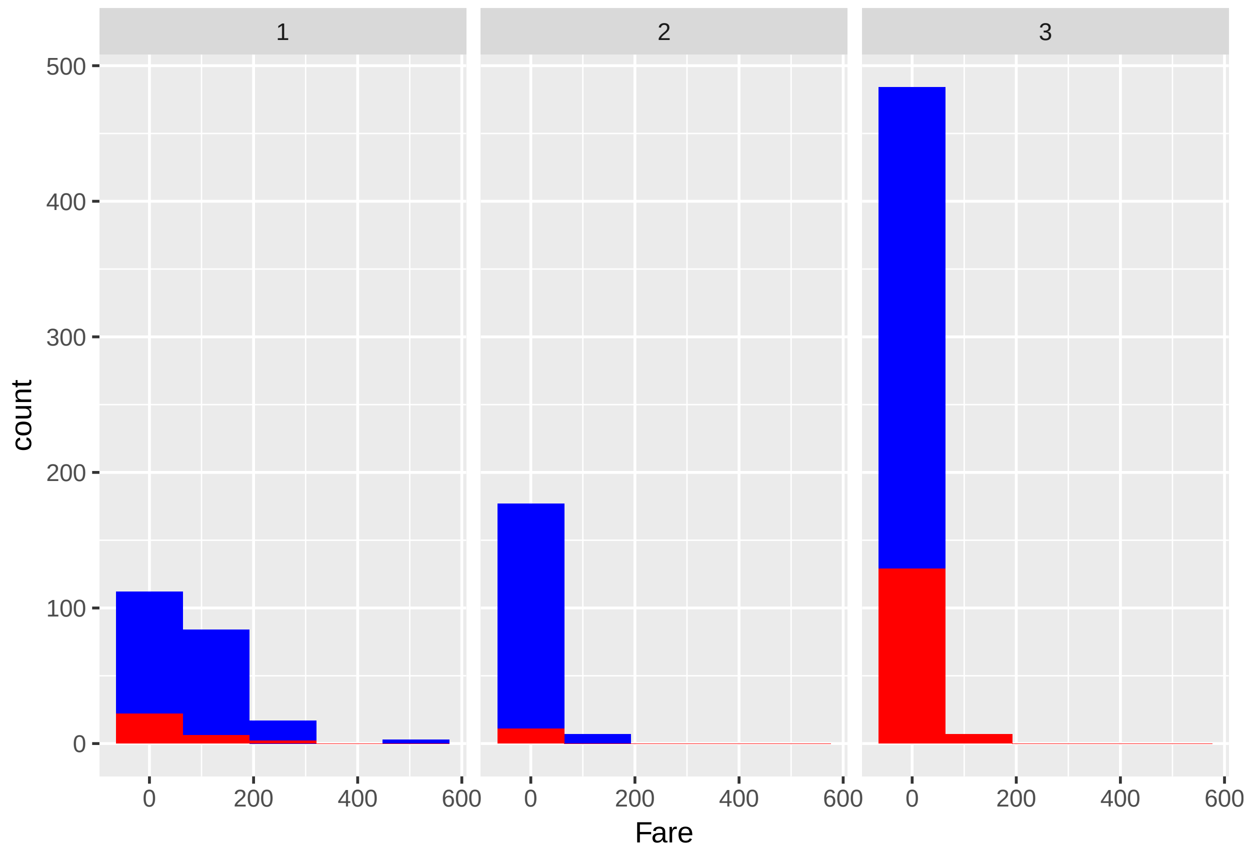
```
##   PassengerId    Survived Pclass      Name               Sex
##  Min.   :  6.0   0:125    1: 30   Length:177         female: 53
##  1st Qu.:230.0   1: 52    2: 11   Class :character   male  :124
##  Median :452.0            3:136   Mode  :character
##  Mean   :435.6
##  3rd Qu.:634.0
##  Max.   :889.0
##
##       Age          SibSp           Parch           Ticket
##  Min.   : NA   Min.   :0.000   Min.   :0.0000   Length:177
##  1st Qu.: NA   1st Qu.:0.000   1st Qu.:0.0000   Class :character
##  Median : NA   Median :0.000   Median :0.0000   Mode  :character
##  Mean   :NaN   Mean   :0.565   Mean   :0.1808
##  3rd Qu.: NA   3rd Qu.:0.000   3rd Qu.:0.0000
##  Max.   : NA   Max.   :8.000   Max.   :2.0000
##  NA's   :177
```

```
##       Fare            Cabin            Embarked
##  Min.   :  0.00   Length:177          : 0
##  1st Qu.:  7.75   Class :character   C:38
##  Median :  8.05   Mode  :character   Q:49
##  Mean   : 22.16                      S:90
##  3rd Qu.: 24.15
##  Max.   :227.53
##
```

Looking on the summary doesn't bring much insight if these passenger do have something in common, other that the missing Age. Thus I will inspect this visually.

```r
library(ggplot2)

ggplot() +
    geom_histogram(data = train, mapping = aes(x = Fare), fill = "blue", bins = 5) +
    geom_histogram(data = missing, mapping = aes(x = Fare), fill = "red", bins = 5) +
    facet_grid(.~ Pclass)
```



From this graph we can notice that most of the passengers with missing Age also have common Pclass value of 3. Here is a summary table of that:

```r
table(missing$Pclass, missing$Sex)
```

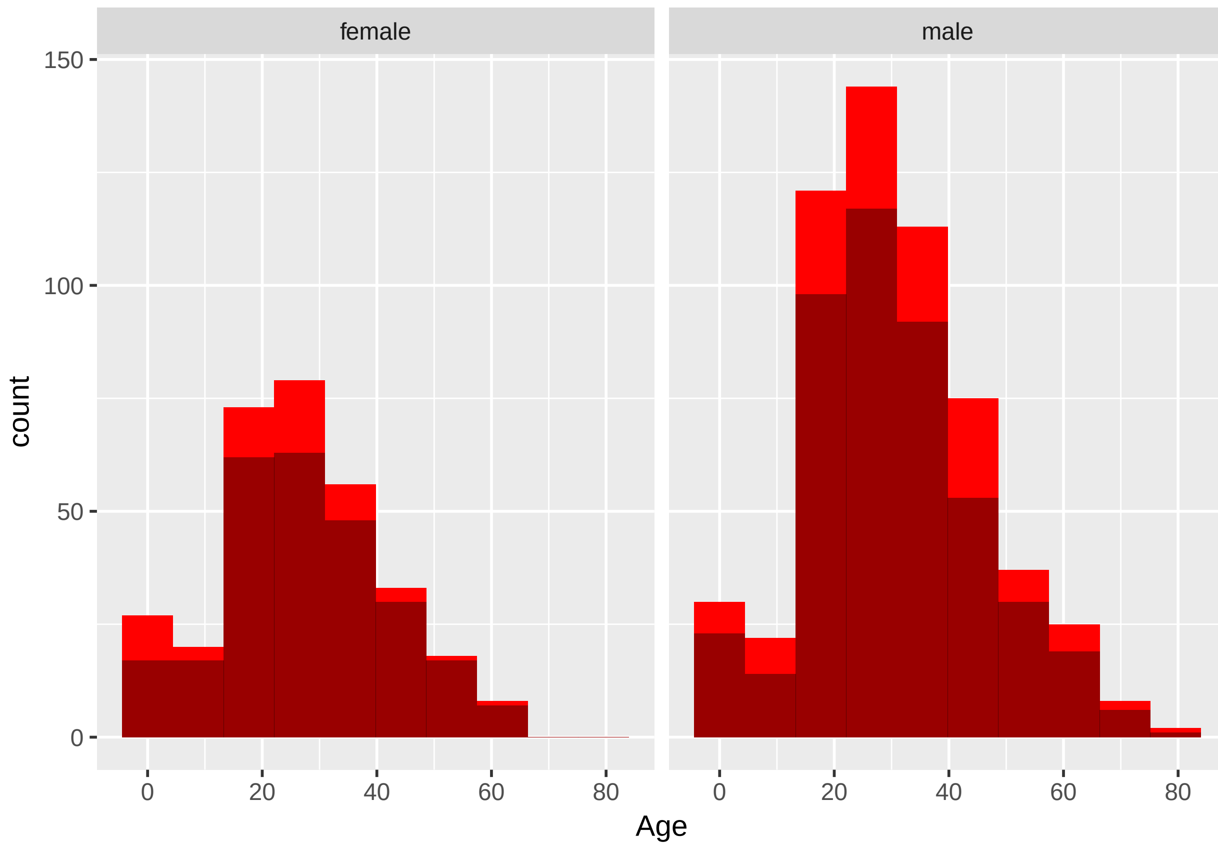
```
##
##     female male
##   1      9   21
##   2      2    9
```

```
## 3     42   94
```

If we simply omit these observations in our logistics regression model we will introduce selection bias in our model as we will remove mostly male, low-fare, low-class passengers. Thus we want to impute the missing data

```
library(mice)
imputed = complete(mice(train,m=5,maxit=50,meth='pmm',seed=500))
imputed.test = complete(mice(test,m=5,maxit=50,meth='pmm',seed=500))
```

```
ggplot() +
    geom_histogram(data = imputed, mapping = aes(x = Age), fill = "red", bins = 10) +
    geom_histogram(data = train, mapping = aes(x = Age), fill = "black", bins = 10, alpha = 0.4) +
    facet_grid(. ~ Sex)
```



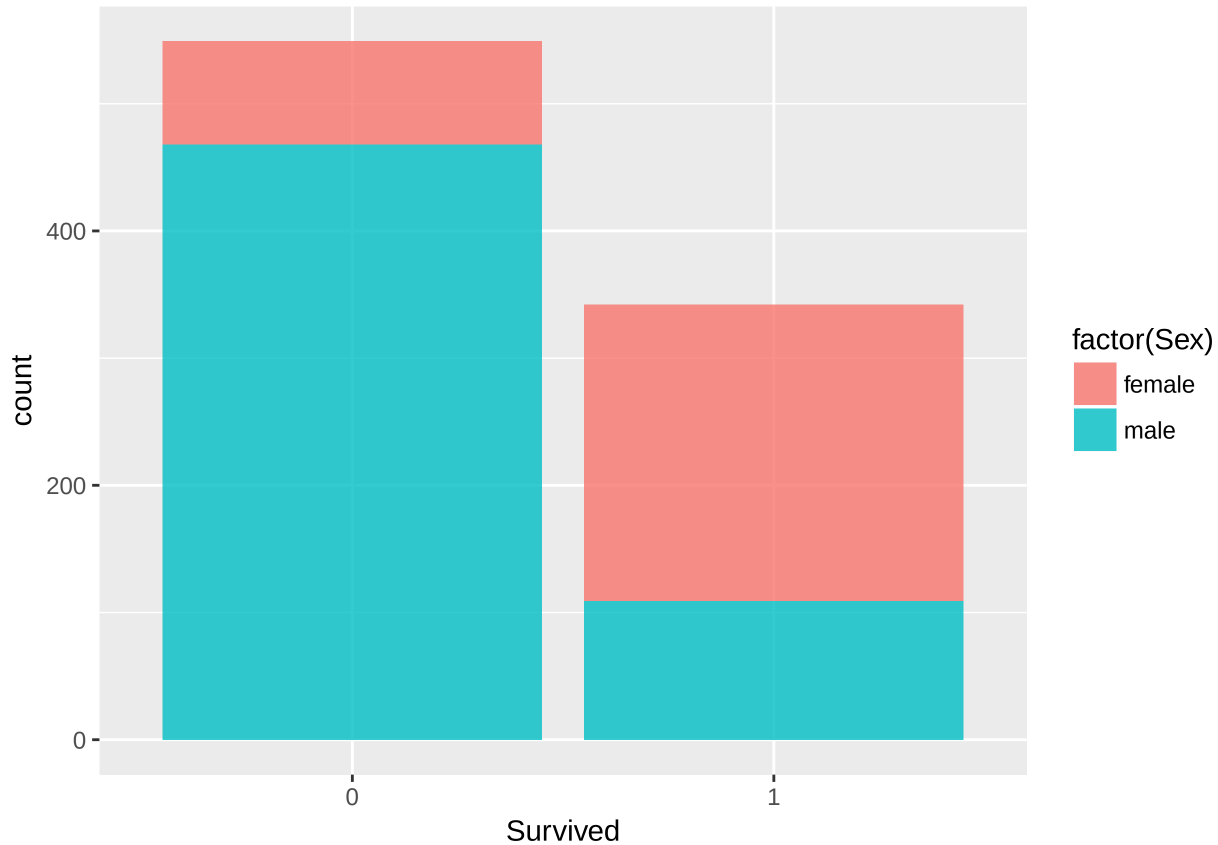
After the imputation I do not have anymore missing data and the Age distribution among women and men seems to be unchanged. Before continuing with my exploratory data analysis I want to construct a dummy baseline model for prediction:

```
table(train$Survived)
```

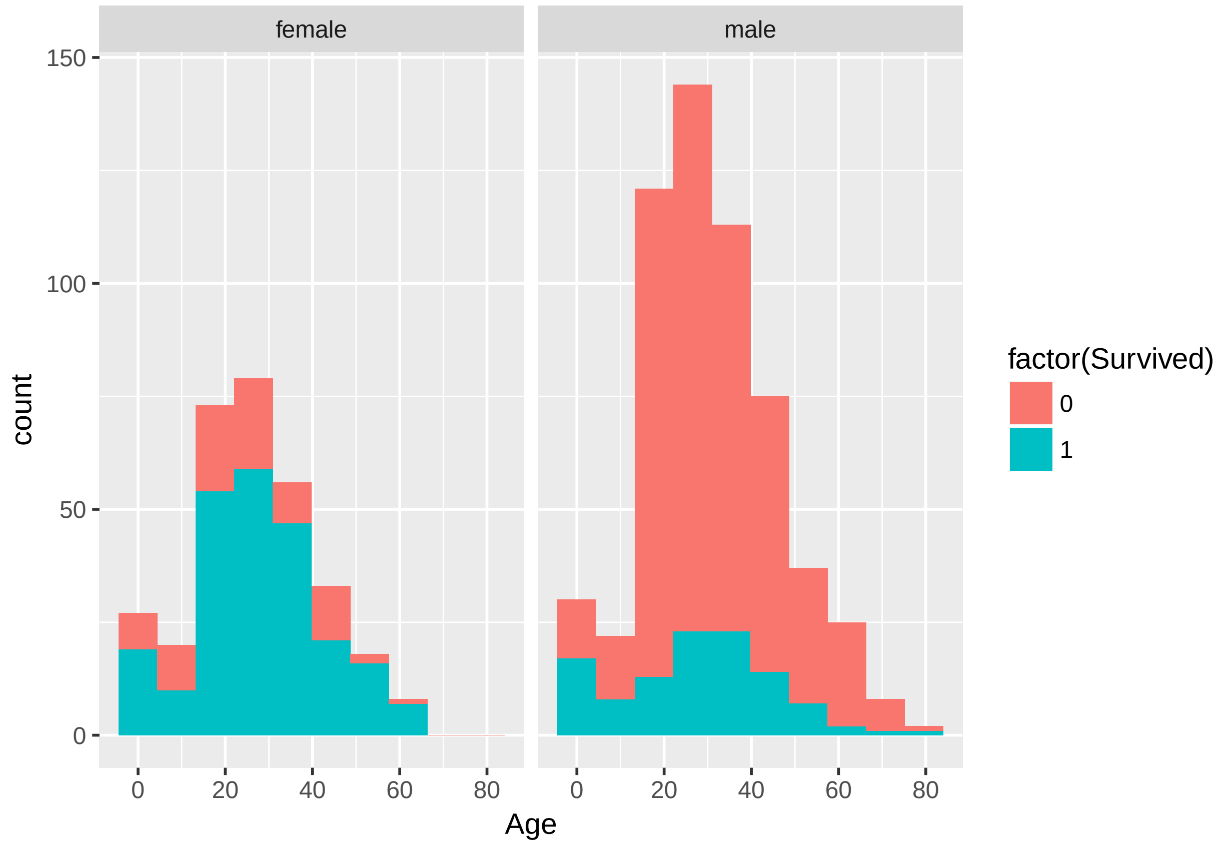
```
##
##   0   1
## 549 342
```

My baseline model will predict that Survived = 0, no matter what and it will be accurate in approximately 62% (549/891) of the cases (which is pretty big accuracy by the way). Thus I need to build a logistic regression model later which can beat that. But before that I need to identify the significant variables which can help to improve my prediction score.

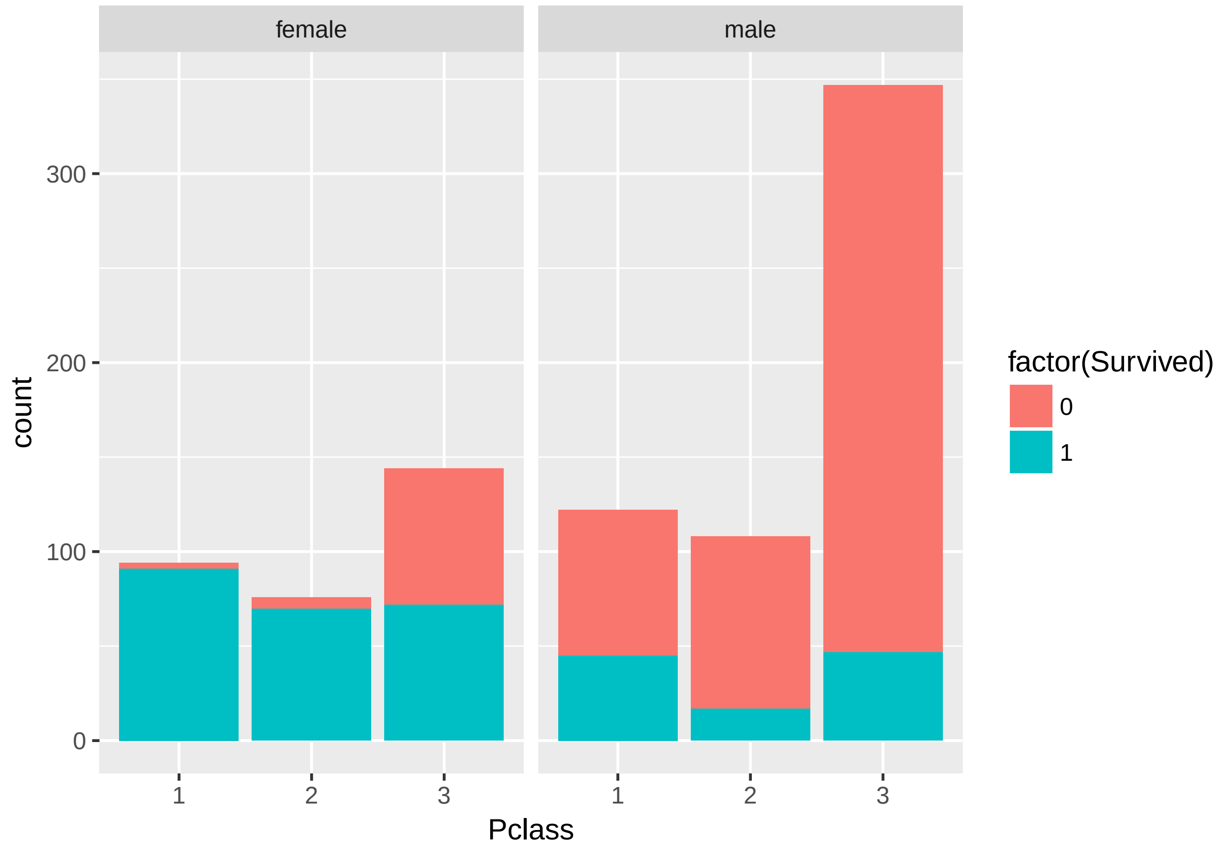
```r
ggplot() +
    geom_histogram(data = imputed, mapping = aes(x = Survived, fill = factor(Sex)), alpha = 0.8, stat="
```



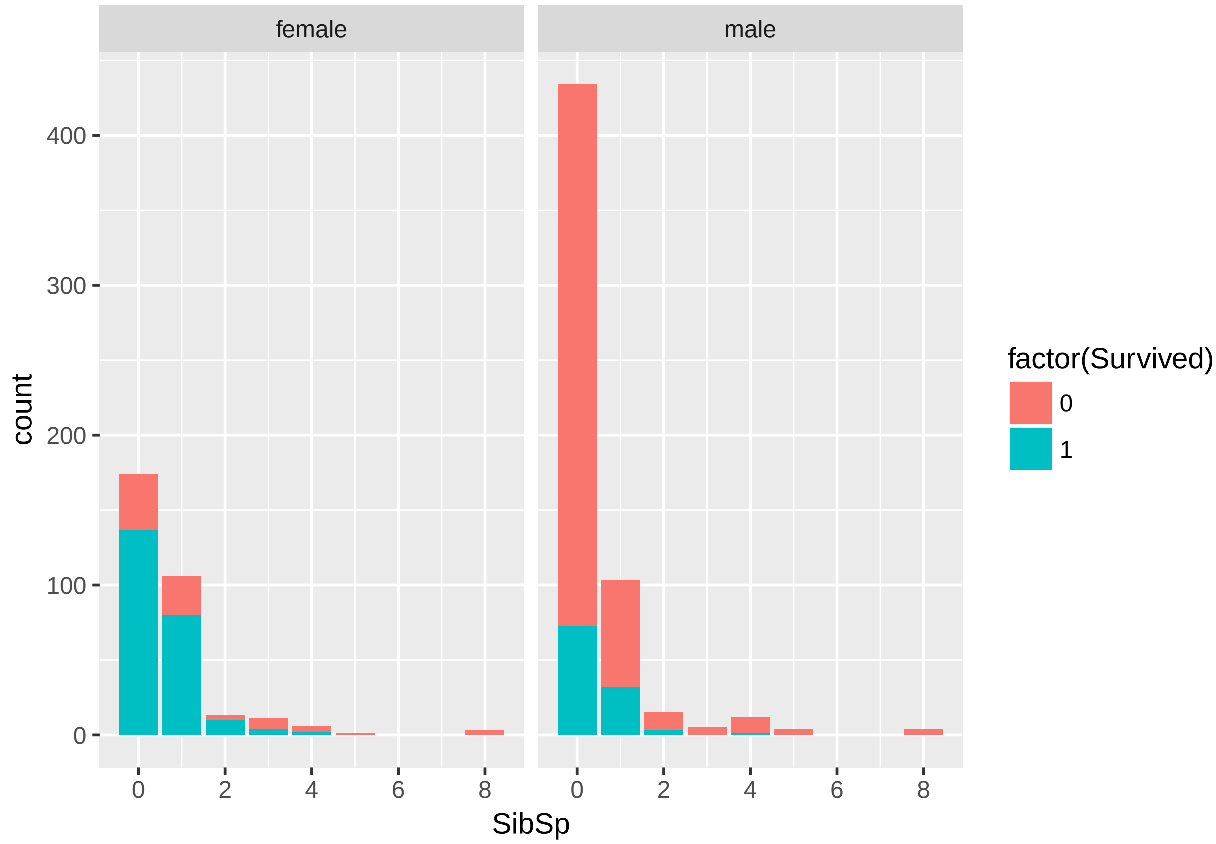

```r
ggplot() +
    geom_histogram(data = imputed, mapping = aes(x = Age, fill = factor(Survived)), bins = 10) +
    facet_grid(. ~ Sex)
```



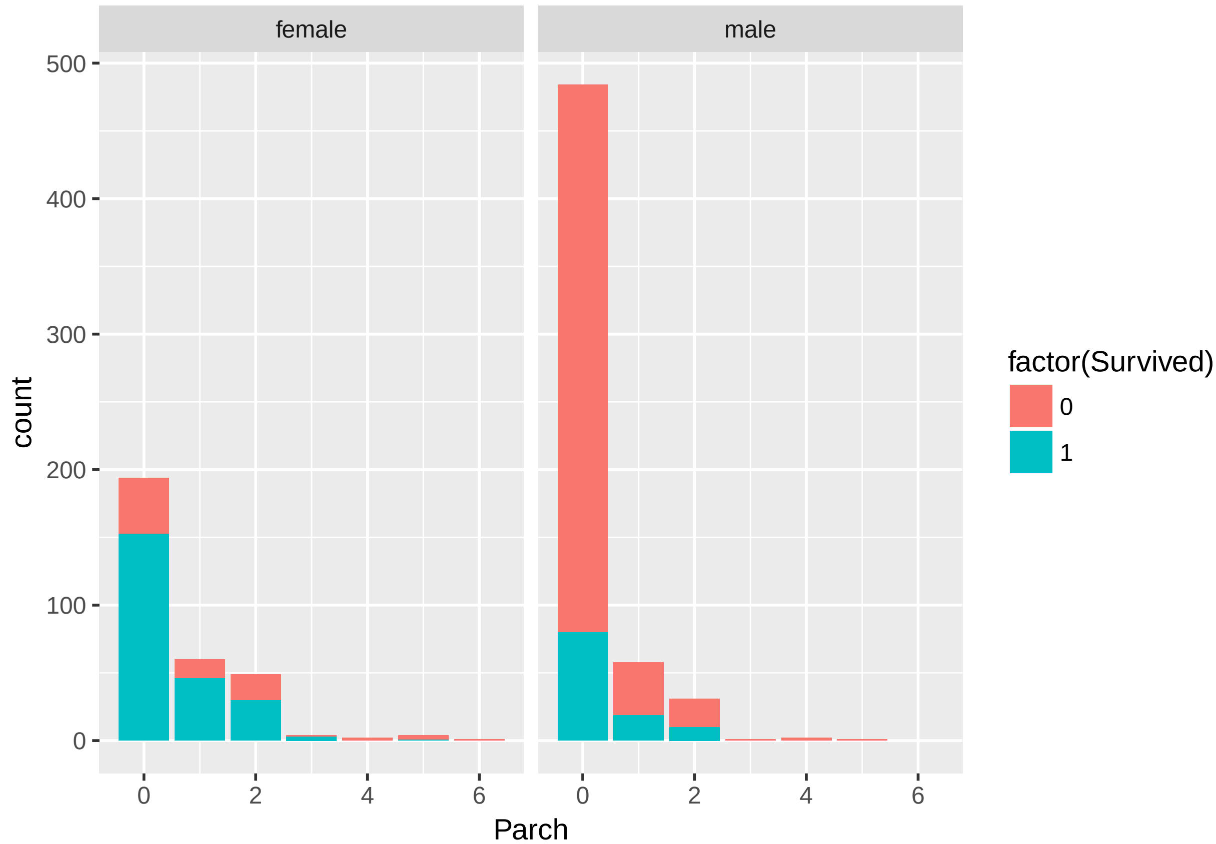

```r
ggplot() +
    geom_histogram(data = imputed, mapping = aes(x = Pclass, fill = factor(Survived)), stat="count") +
    facet_grid(. ~ Sex)
```



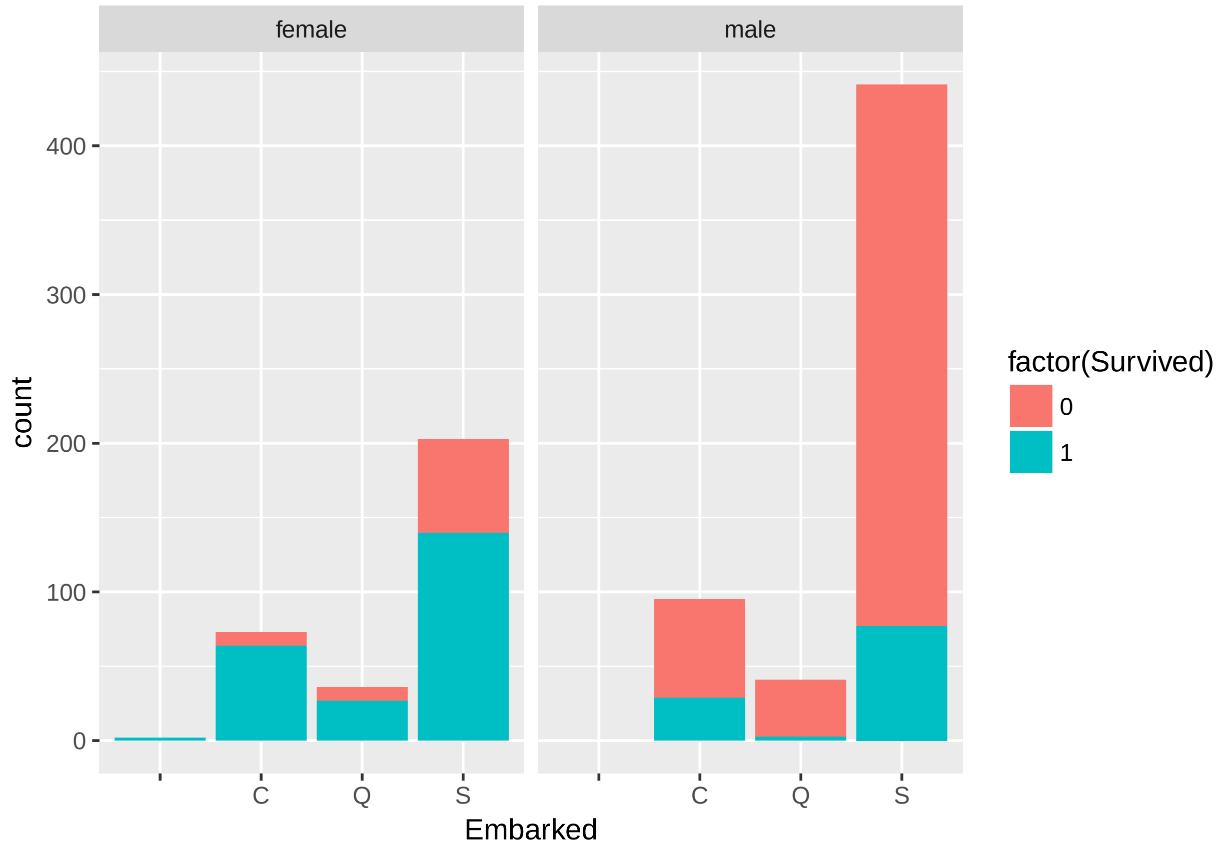

```
ggplot() +
    geom_histogram(data = imputed, mapping = aes(x = SibSp, fill = factor(Survived)), stat="count") +
    facet_grid(. ~ Sex)
```




```
ggplot() +
    geom_histogram(data = imputed, mapping = aes(x = Parch, fill = factor(Survived)), stat="count") +
    facet_grid(. ~ Sex)
```




```
ggplot() +
    geom_histogram(data = imputed, mapping = aes(x = Embarked, fill = factor(Survived)), stat="count") +
    facet_grid(. ~ Sex)
```



Building my first logistic regression model

```
imputed = subset(imputed, select = -c(PassengerId))
model1 = glm(Survived ~ Sex + Pclass + Embarked + SibSp, data = imputed, family=binomial)
predict1 = predict(model1, type="response")

a = table(imputed$Survived, predict1 >= 0.5)

TP = a[2,2] # true positives
TN = a[1,1] # true negatives
FP = a[1,2] # false positives
FN = a[2,1] # false negatives

sensitivity = TP/(TP+FN)
specificity = TN/(TN+FN)
accuracy = (TN + TP)/(TN + TP + FP + FN)
```

The accuracy of this model is 79.2% on the training set. Let's see what's the AUC value:

```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

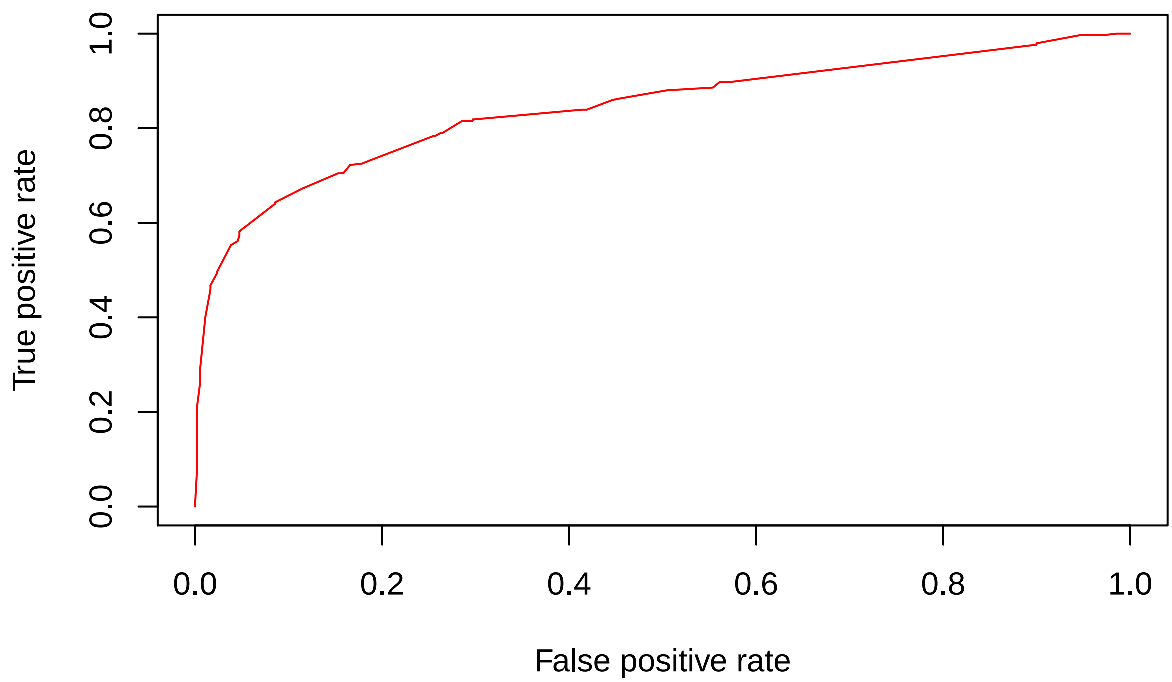
```
## The following object is masked from 'package:stats':
##
##     lowess
```

```r
ROCRpred = prediction(predict1, train$Survived)
as.numeric(performance(ROCRpred, "auc")@y.values)
```

```
## [1] 0.841578
```

Well this AUC value is quite high, which means that my model quite differentiate between the passengers who survived and those who didn't.

```r
perf = performance(ROCRpred,measure = "tpr", x.measure = "fpr")
plot(perf, col=rainbow(10))
```



Logistic Regression using feature engineering

Well my logistic regression model works a bit better than the baseline model but I believe it can do much better if I apply feature engineering. But first I will copy the missing age observations to my training set:

```r
train$Age = imputed$Age
test$Age = imputed.test$Age
```

First I will start by adding a factor variable indicating of the passenger is a child:

```r
train$child = train$Age < 18
test$child = test$Age < 18
```

Next I will add a numerical variable showing the number of accompanying family members:

```r
train$familySize = train$SibSp + train$Parch + 1
test$familySize = test$SibSp + test$Parch + 1
```

Now I will separate the title from the passengers' names and put it in an additional variable:

```
train$Title = gsub('(.*, )|(\\..*)', '', train$Name)
test$Title = gsub('(.*, )|(\\..*)', '', test$Name)
```

```
# code taken 1:1 from Megan Risdal

rare_title <- c('Dona', 'Lady', 'the Countess','Capt', 'Col', 'Don',
                'Dr', 'Major', 'Rev', 'Sir', 'Jonkheer')

train$Title[train$Title == 'Mlle']        <- 'Miss'
train$Title[train$Title == 'Ms']          <- 'Miss'
train$Title[train$Title == 'Mme']         <- 'Mrs'
train$Title[train$Title %in% rare_title]  <- 'Rare Title'

test$Title[test$Title == 'Mlle']        <- 'Miss'
test$Title[test$Title == 'Ms']          <- 'Miss'
test$Title[test$Title == 'Mme']         <- 'Mrs'
test$Title[test$Title %in% rare_title]  <- 'Rare Title'
```

```
train$mother =  train$Sex == "female" & train$Parch > 0 & train$child == FALSE & train$Title != "Miss"
test$mother = test$Sex == 'female' & test$Parch > 0 & test$child == FALSE & test$Title != "Miss"
```

Now I will build my second logistic regression model:

```
data2 = subset(train, select = -c(PassengerId))
model2 = glm(Survived ~ Sex + Pclass + Embarked + SibSp + mother + child + Title + familySize, data = da
predict2 = predict(model2, type="response")

a = table(data2$Survived, predict2 >= 0.5)

TP = a[2,2] # true positives
TN = a[1,1] # true negatives
FP = a[1,2] # false positives
FN = a[2,1] # false negatives

sensitivity = TP/(TP+FN)
specificity = TN/(TN+FN)
accuracy = (TN + TP)/(TN + TP + FP + FN)
```

The accuracy on the training set increased to 83.16%. Now regarding the new AUC score:

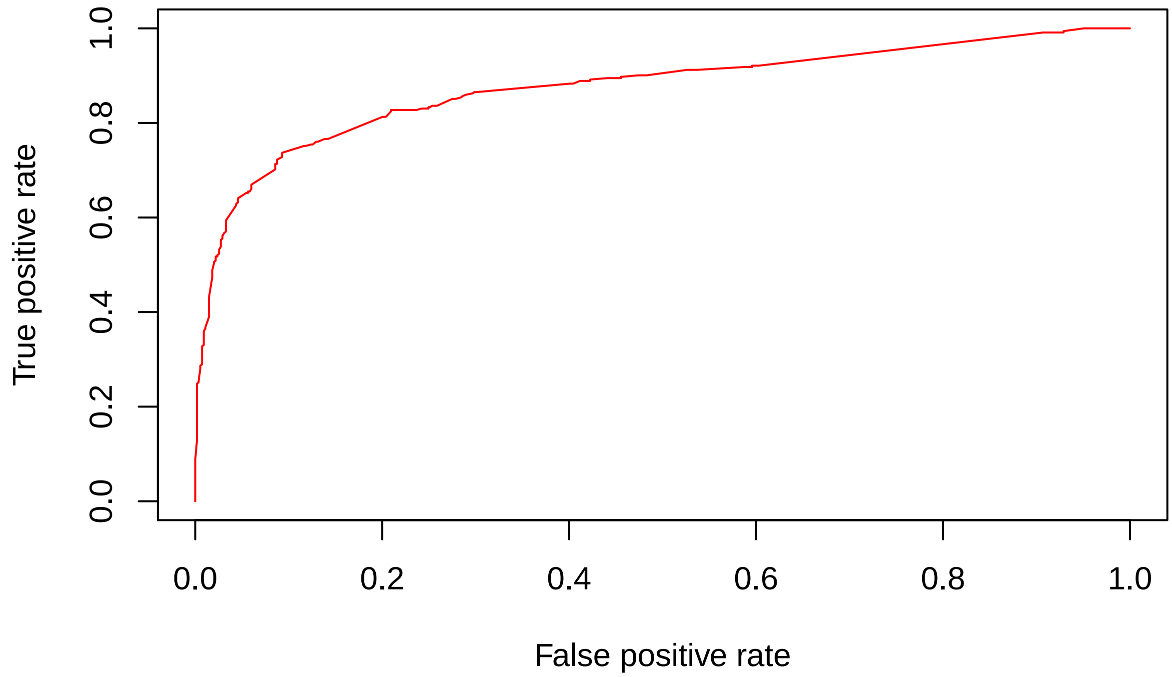
```
library(ROCR)

ROCRpred = prediction(predict2, data2$Survived)
as.numeric(performance(ROCRpred, "auc")@y.values)
```

```
## [1] 0.8746658
```

Well this AUC value is quite high, which means that my model quite differentiate between the passengers who survived and those who didn't.

```
perf2 = performance(ROCRpred,measure = "tpr", x.measure = "fpr")
plot(perf2, col=rainbow(10))
```



Let's submit this version and see how it will get scored in Kaggle.

```
prediction2 <- predict(model2, newdata=test, type = "response")
solution2 <- data.frame(PassengerID = test$PassengerId, Survived = round(prediction2, 0))
write.csv(solution2, file = 'model2_Solution.csv', row.names = F)
```

This model was scored 77.99% at Kaggle.

Random Forest using feature engineering

Now I want to test how a non-linear model will score on this problem. For doing this I will use a random forest to determine which would be the best regression tree for this dataset.

```
library(randomForest)
```

```
## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rpart.plot)
```

```
## Loading required package: rpart
```

```r
library(rpart)
library(caret)
```

```
## Loading required package: lattice
```

```r
library(e1071)
```

```r
# Defining cross-validation experiment
numFolds = trainControl(method = "cv", number = 50 )
cpGrid = expand.grid( .cp = seq(0.005,0.05,0.0001))

# Performing the cross validation
train(Survived ~ Sex +
                 Pclass +
                 Embarked +
                 SibSp +
                 mother +
                 child +
                 Title +
                 familySize, data = train, method = "rpart", trControl = numFolds, tuneGrid = cpGrid )
```

```
## CART
##
## 891 samples
##   8 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (50 fold)
## Summary of sample sizes: 873, 873, 873, 873, 874, 873, ...
## Resampling results across tuning parameters:
##
##   cp      Accuracy   Kappa
##   0.0050  0.8026797  0.5639275
##   0.0051  0.8026797  0.5639275
##   0.0052  0.8026797  0.5639275
##   0.0053  0.8026797  0.5639275
##   0.0054  0.8026797  0.5639275
##   0.0055  0.8026797  0.5639275
##   0.0056  0.8026797  0.5639275
##   0.0057  0.8026797  0.5639275
##   0.0058  0.8026797  0.5639275
##   0.0059  0.8026797  0.5639275
##   0.0060  0.8015686  0.5625960
##   0.0061  0.8015686  0.5625960
##   0.0062  0.8015686  0.5625960
##   0.0063  0.8015686  0.5625960
##   0.0064  0.8015686  0.5625960
##   0.0065  0.8015686  0.5625960
##   0.0066  0.8015686  0.5625960
##   0.0067  0.8015686  0.5625960
##   0.0068  0.8015686  0.5625960
##   0.0069  0.8015686  0.5625960
##   0.0070  0.8015686  0.5633645
##   0.0071  0.8015686  0.5633645
```

```
## 0.0072 0.8015686 0.5633645
## 0.0073 0.8015686 0.5633645
## 0.0074 0.8015686 0.5633645
## 0.0075 0.8128105 0.5924226
## 0.0076 0.8128105 0.5924226
## 0.0077 0.8128105 0.5924226
## 0.0078 0.8128105 0.5924226
## 0.0079 0.8128105 0.5924226
## 0.0080 0.8128105 0.5924226
## 0.0081 0.8128105 0.5924226
## 0.0082 0.8128105 0.5924226
## 0.0083 0.8128105 0.5924226
## 0.0084 0.8128105 0.5924226
## 0.0085 0.8128105 0.5924226
## 0.0086 0.8128105 0.5924226
## 0.0087 0.8128105 0.5924226
## 0.0088 0.8128105 0.5924226
## 0.0089 0.8128105 0.5924226
## 0.0090 0.8151634 0.6021701
## 0.0091 0.8151634 0.6021701
## 0.0092 0.8151634 0.6021701
## 0.0093 0.8151634 0.6021701
## 0.0094 0.8151634 0.6021701
## 0.0095 0.8151634 0.6021701
## 0.0096 0.8151634 0.6021701
## 0.0097 0.8151634 0.6021701
## 0.0098 0.8151634 0.6021701
## 0.0099 0.8151634 0.6021701
## 0.0100 0.8162745 0.6046736
## 0.0101 0.8162745 0.6046736
## 0.0102 0.8162745 0.6046736
## 0.0103 0.8162745 0.6046736
## 0.0104 0.8162745 0.6046736
## 0.0105 0.8162745 0.6046736
## 0.0106 0.8162745 0.6046736
## 0.0107 0.8162745 0.6046736
## 0.0108 0.8162745 0.6046736
## 0.0109 0.8162745 0.6046736
## 0.0110 0.8162745 0.6046736
## 0.0111 0.8162745 0.6046736
## 0.0112 0.8162745 0.6046736
## 0.0113 0.8162745 0.6046736
## 0.0114 0.8162745 0.6046736
## 0.0115 0.8162745 0.6046736
## 0.0116 0.8162745 0.6046736
## 0.0117 0.8162745 0.6046736
## 0.0118 0.8162745 0.6046736
## 0.0119 0.8162745 0.6046736
## 0.0120 0.8173856 0.6073477
## 0.0121 0.8173856 0.6073477
## 0.0122 0.8173856 0.6073477
## 0.0123 0.8173856 0.6073477
## 0.0124 0.8173856 0.6073477
## 0.0125 0.8173856 0.6073477
```

```
## 0.0126 0.8173856 0.6073477
## 0.0127 0.8173856 0.6073477
## 0.0128 0.8173856 0.6073477
## 0.0129 0.8173856 0.6073477
## 0.0130 0.8173856 0.6073477
## 0.0131 0.8173856 0.6073477
## 0.0132 0.8173856 0.6073477
## 0.0133 0.8173856 0.6073477
## 0.0134 0.8173856 0.6073477
## 0.0135 0.8173856 0.6073477
## 0.0136 0.8173856 0.6073477
## 0.0137 0.8173856 0.6073477
## 0.0138 0.8173856 0.6073477
## 0.0139 0.8173856 0.6073477
## 0.0140 0.8173856 0.6073477
## 0.0141 0.8173856 0.6073477
## 0.0142 0.8173856 0.6073477
## 0.0143 0.8173856 0.6073477
## 0.0144 0.8173856 0.6073477
## 0.0145 0.8173856 0.6073477
## 0.0146 0.8173856 0.6073477
## 0.0147 0.8173856 0.6073477
## 0.0148 0.8173856 0.6073477
## 0.0149 0.8150327 0.6030407
## 0.0150 0.8150327 0.6030407
## 0.0151 0.8150327 0.6030407
## 0.0152 0.8150327 0.6030407
## 0.0153 0.8150327 0.6030407
## 0.0154 0.8150327 0.6030407
## 0.0155 0.8150327 0.6030407
## 0.0156 0.8150327 0.6030407
## 0.0157 0.8150327 0.6030407
## 0.0158 0.8150327 0.6030407
## 0.0159 0.8150327 0.6030407
## 0.0160 0.8150327 0.6030407
## 0.0161 0.8150327 0.6030407
## 0.0162 0.8150327 0.6030407
## 0.0163 0.8150327 0.6030407
## 0.0164 0.8150327 0.6030407
## 0.0165 0.8150327 0.6030407
## 0.0166 0.8150327 0.6030407
## 0.0167 0.8150327 0.6030407
## 0.0168 0.8150327 0.6030407
## 0.0169 0.8150327 0.6030407
## 0.0170 0.8150327 0.6030407
## 0.0171 0.8150327 0.6030407
## 0.0172 0.8150327 0.6030407
## 0.0173 0.8150327 0.6030407
## 0.0174 0.8150327 0.6030407
## 0.0175 0.8150327 0.6030407
## 0.0176 0.8150327 0.6030407
## 0.0177 0.8150327 0.6030407
## 0.0178 0.8150327 0.6030407
## 0.0179 0.8138562 0.6005664
```

```
## 0.0180 0.8105229 0.5939618
## 0.0181 0.8105229 0.5939618
## 0.0182 0.8105229 0.5939618
## 0.0183 0.8105229 0.5939618
## 0.0184 0.8105229 0.5939618
## 0.0185 0.8105229 0.5939618
## 0.0186 0.8105229 0.5939618
## 0.0187 0.8105229 0.5939618
## 0.0188 0.8105229 0.5939618
## 0.0189 0.8105229 0.5939618
## 0.0190 0.8071895 0.5859270
## 0.0191 0.8071895 0.5859270
## 0.0192 0.8071895 0.5859270
## 0.0193 0.8071895 0.5859270
## 0.0194 0.8071895 0.5859270
## 0.0195 0.8071895 0.5859270
## 0.0196 0.8071895 0.5859270
## 0.0197 0.8071895 0.5859270
## 0.0198 0.8071895 0.5859270
## 0.0199 0.8071895 0.5859270
## 0.0200 0.8071895 0.5859270
## 0.0201 0.8071895 0.5859270
## 0.0202 0.8071895 0.5859270
## 0.0203 0.8071895 0.5859270
## 0.0204 0.8071895 0.5859270
## 0.0205 0.8071895 0.5859270
## 0.0206 0.8071895 0.5859270
## 0.0207 0.8071895 0.5859270
## 0.0208 0.8071895 0.5859270
## 0.0209 0.8060784 0.5838827
## 0.0210 0.8060784 0.5838827
## 0.0211 0.8060784 0.5838827
## 0.0212 0.8060784 0.5838827
## 0.0213 0.8060784 0.5838827
## 0.0214 0.8060784 0.5838827
## 0.0215 0.8060784 0.5838827
## 0.0216 0.8060784 0.5838827
## 0.0217 0.8060784 0.5838827
## 0.0218 0.8060784 0.5838827
## 0.0219 0.8060784 0.5838827
## 0.0220 0.8060784 0.5838827
## 0.0221 0.8060784 0.5838827
## 0.0222 0.8060784 0.5838827
## 0.0223 0.8060784 0.5838827
## 0.0224 0.8060784 0.5838827
## 0.0225 0.8060784 0.5838827
## 0.0226 0.8060784 0.5838827
## 0.0227 0.8060784 0.5838827
## 0.0228 0.8060784 0.5838827
## 0.0229 0.8060784 0.5838827
## 0.0230 0.8060784 0.5838827
## 0.0231 0.8060784 0.5838827
## 0.0232 0.8060784 0.5838827
## 0.0233 0.8060784 0.5838827
```

```
##   0.0234  0.8060784  0.5838827
##   0.0235  0.8060784  0.5838827
##   0.0236  0.8060784  0.5838827
##   0.0237  0.8060784  0.5838827
##   0.0238  0.8060784  0.5838827
##   0.0239  0.8116993  0.5969313
##   0.0240  0.8116993  0.5969313
##   0.0241  0.8116993  0.5969313
##   0.0242  0.8116993  0.5969313
##   0.0243  0.8116993  0.5969313
##   0.0244  0.8116993  0.5969313
##   0.0245  0.8116993  0.5969313
##   0.0246  0.8116993  0.5969313
##   0.0247  0.8116993  0.5969313
##   0.0248  0.8116993  0.5969313
##   0.0249  0.8116993  0.5969313
##   0.0250  0.8116993  0.5969313
##   0.0251  0.8116993  0.5969313
##   0.0252  0.8116993  0.5969313
##   0.0253  0.8116993  0.5969313
##   0.0254  0.8116993  0.5969313
##   0.0255  0.8116993  0.5969313
##   0.0256  0.8116993  0.5969313
##   0.0257  0.8116993  0.5969313
##   0.0258  0.8116993  0.5969313
##   0.0259  0.8116993  0.5969313
##   0.0260  0.8116993  0.5969313
##   0.0261  0.8116993  0.5969313
##   0.0262  0.8116993  0.5969313
##   0.0263  0.8116993  0.5969313
##   0.0264  0.8116993  0.5969313
##   0.0265  0.8116993  0.5969313
##   0.0266  0.8116993  0.5969313
##   0.0267  0.8116993  0.5969313
##   0.0268  0.8116993  0.5969313
##   0.0269  0.8116993  0.5969313
##   0.0270  0.8116993  0.5969313
##   0.0271  0.8116993  0.5969313
##   0.0272  0.8116993  0.5969313
##   0.0273  0.8116993  0.5969313
##   0.0274  0.8116993  0.5969313
##   0.0275  0.8116993  0.5969313
##   0.0276  0.8116993  0.5969313
##   0.0277  0.8116993  0.5969313
##   0.0278  0.8116993  0.5969313
##   0.0279  0.8116993  0.5969313
##   0.0280  0.8116993  0.5969313
##   0.0281  0.8116993  0.5969313
##   0.0282  0.8116993  0.5969313
##   0.0283  0.8116993  0.5969313
##   0.0284  0.8116993  0.5969313
##   0.0285  0.8116993  0.5969313
##   0.0286  0.8116993  0.5969313
##   0.0287  0.8116993  0.5969313
```

```
## 0.0288 0.8116993 0.5969313
## 0.0289 0.8116993 0.5969313
## 0.0290 0.8116993 0.5969313
## 0.0291 0.8116993 0.5969313
## 0.0292 0.8116993 0.5969313
## 0.0293 0.8116993 0.5969313
## 0.0294 0.8116993 0.5969313
## 0.0295 0.8116993 0.5969313
## 0.0296 0.8116993 0.5969313
## 0.0297 0.8116993 0.5969313
## 0.0298 0.8116993 0.5969313
## 0.0299 0.8094771 0.5928616
## 0.0300 0.8094771 0.5928616
## 0.0301 0.8094771 0.5928616
## 0.0302 0.8094771 0.5928616
## 0.0303 0.8094771 0.5928616
## 0.0304 0.8094771 0.5928616
## 0.0305 0.8094771 0.5928616
## 0.0306 0.8094771 0.5928616
## 0.0307 0.8094771 0.5928616
## 0.0308 0.8094771 0.5928616
## 0.0309 0.8094771 0.5928616
## 0.0310 0.8094771 0.5928616
## 0.0311 0.8094771 0.5928616
## 0.0312 0.8094771 0.5928616
## 0.0313 0.8094771 0.5928616
## 0.0314 0.8094771 0.5928616
## 0.0315 0.8094771 0.5928616
## 0.0316 0.8094771 0.5928616
## 0.0317 0.8094771 0.5928616
## 0.0318 0.8094771 0.5928616
## 0.0319 0.8094771 0.5928616
## 0.0320 0.8094771 0.5928616
## 0.0321 0.8094771 0.5928616
## 0.0322 0.8094771 0.5928616
## 0.0323 0.8094771 0.5928616
## 0.0324 0.8094771 0.5928616
## 0.0325 0.8094771 0.5928616
## 0.0326 0.8094771 0.5928616
## 0.0327 0.8094771 0.5928616
## 0.0328 0.8094771 0.5928616
## 0.0329 0.8094771 0.5928616
## 0.0330 0.8094771 0.5928616
## 0.0331 0.8094771 0.5928616
## 0.0332 0.8094771 0.5928616
## 0.0333 0.8094771 0.5928616
## 0.0334 0.8094771 0.5928616
## 0.0335 0.8094771 0.5928616
## 0.0336 0.8094771 0.5928616
## 0.0337 0.8094771 0.5928616
## 0.0338 0.8094771 0.5928616
## 0.0339 0.8094771 0.5928616
## 0.0340 0.8094771 0.5928616
## 0.0341 0.8094771 0.5928616
```

```
## 0.0342 0.8094771 0.5928616
## 0.0343 0.8094771 0.5928616
## 0.0344 0.8094771 0.5928616
## 0.0345 0.8094771 0.5928616
## 0.0346 0.8094771 0.5928616
## 0.0347 0.8094771 0.5928616
## 0.0348 0.8094771 0.5928616
## 0.0349 0.8094771 0.5928616
## 0.0350 0.8094771 0.5928616
## 0.0351 0.8094771 0.5928616
## 0.0352 0.8094771 0.5928616
## 0.0353 0.8094771 0.5928616
## 0.0354 0.8094771 0.5928616
## 0.0355 0.8094771 0.5928616
## 0.0356 0.8094771 0.5928616
## 0.0357 0.8094771 0.5928616
## 0.0358 0.8094771 0.5928616
## 0.0359 0.8094771 0.5928616
## 0.0360 0.8094771 0.5928616
## 0.0361 0.8094771 0.5928616
## 0.0362 0.8094771 0.5928616
## 0.0363 0.8094771 0.5928616
## 0.0364 0.8094771 0.5928616
## 0.0365 0.8094771 0.5928616
## 0.0366 0.8094771 0.5928616
## 0.0367 0.8094771 0.5928616
## 0.0368 0.8094771 0.5928616
## 0.0369 0.8094771 0.5928616
## 0.0370 0.8094771 0.5928616
## 0.0371 0.8094771 0.5928616
## 0.0372 0.8094771 0.5928616
## 0.0373 0.8094771 0.5928616
## 0.0374 0.8094771 0.5928616
## 0.0375 0.8094771 0.5928616
## 0.0376 0.8094771 0.5928616
## 0.0377 0.8094771 0.5928616
## 0.0378 0.8094771 0.5928616
## 0.0379 0.8094771 0.5928616
## 0.0380 0.8094771 0.5928616
## 0.0381 0.8094771 0.5928616
## 0.0382 0.8094771 0.5928616
## 0.0383 0.8094771 0.5928616
## 0.0384 0.8094771 0.5928616
## 0.0385 0.8094771 0.5928616
## 0.0386 0.8094771 0.5928616
## 0.0387 0.8094771 0.5928616
## 0.0388 0.8094771 0.5928616
## 0.0389 0.8094771 0.5928616
## 0.0390 0.8094771 0.5928616
## 0.0391 0.8094771 0.5928616
## 0.0392 0.8094771 0.5928616
## 0.0393 0.8094771 0.5928616
## 0.0394 0.8094771 0.5928616
## 0.0395 0.8094771 0.5928616
```

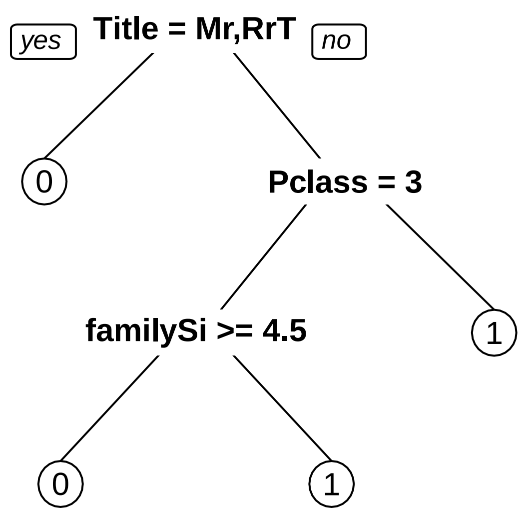
```
## 0.0396 0.8094771 0.5928616
## 0.0397 0.8094771 0.5928616
## 0.0398 0.8094771 0.5928616
## 0.0399 0.8094771 0.5928616
## 0.0400 0.8094771 0.5928616
## 0.0401 0.8094771 0.5928616
## 0.0402 0.8094771 0.5928616
## 0.0403 0.8094771 0.5928616
## 0.0404 0.8094771 0.5928616
## 0.0405 0.8094771 0.5928616
## 0.0406 0.8094771 0.5928616
## 0.0407 0.8094771 0.5928616
## 0.0408 0.8094771 0.5928616
## 0.0409 0.8094771 0.5928616
## 0.0410 0.8094771 0.5928616
## 0.0411 0.8094771 0.5928616
## 0.0412 0.8094771 0.5928616
## 0.0413 0.8094771 0.5928616
## 0.0414 0.8094771 0.5928616
## 0.0415 0.8094771 0.5928616
## 0.0416 0.8094771 0.5928616
## 0.0417 0.8094771 0.5928616
## 0.0418 0.8094771 0.5928616
## 0.0419 0.8094771 0.5928616
## 0.0420 0.8094771 0.5928616
## 0.0421 0.8094771 0.5928616
## 0.0422 0.8094771 0.5928616
## 0.0423 0.8094771 0.5928616
## 0.0424 0.8094771 0.5928616
## 0.0425 0.8094771 0.5928616
## 0.0426 0.8094771 0.5928616
## 0.0427 0.8094771 0.5928616
## 0.0428 0.8094771 0.5928616
## 0.0429 0.8094771 0.5928616
## 0.0430 0.8094771 0.5928616
## 0.0431 0.8094771 0.5928616
## 0.0432 0.8094771 0.5928616
## 0.0433 0.8094771 0.5928616
## 0.0434 0.8094771 0.5928616
## 0.0435 0.8094771 0.5928616
## 0.0436 0.8094771 0.5928616
## 0.0437 0.8094771 0.5928616
## 0.0438 0.8094771 0.5928616
## 0.0439 0.8094771 0.5928616
## 0.0440 0.8094771 0.5928616
## 0.0441 0.8094771 0.5928616
## 0.0442 0.8094771 0.5928616
## 0.0443 0.8094771 0.5928616
## 0.0444 0.8094771 0.5928616
## 0.0445 0.8094771 0.5928616
## 0.0446 0.8094771 0.5928616
## 0.0447 0.8094771 0.5928616
## 0.0448 0.8094771 0.5928616
## 0.0449 0.8094771 0.5928616
```

```
##   0.0450  0.8094771  0.5928616
##   0.0451  0.8094771  0.5928616
##   0.0452  0.8094771  0.5928616
##   0.0453  0.8094771  0.5928616
##   0.0454  0.8094771  0.5928616
##   0.0455  0.8094771  0.5928616
##   0.0456  0.8094771  0.5928616
##   0.0457  0.8094771  0.5928616
##   0.0458  0.8094771  0.5928616
##   0.0459  0.8094771  0.5928616
##   0.0460  0.8094771  0.5928616
##   0.0461  0.8094771  0.5928616
##   0.0462  0.8094771  0.5928616
##   0.0463  0.8094771  0.5928616
##   0.0464  0.8094771  0.5928616
##   0.0465  0.8094771  0.5928616
##   0.0466  0.8094771  0.5928616
##   0.0467  0.8094771  0.5928616
##   0.0468  0.8094771  0.5928616
##   0.0469  0.8094771  0.5928616
##   0.0470  0.8094771  0.5928616
##   0.0471  0.8094771  0.5928616
##   0.0472  0.8094771  0.5928616
##   0.0473  0.8094771  0.5928616
##   0.0474  0.8094771  0.5928616
##   0.0475  0.8094771  0.5928616
##   0.0476  0.8094771  0.5928616
##   0.0477  0.8094771  0.5928616
##   0.0478  0.8094771  0.5928616
##   0.0479  0.8094771  0.5928616
##   0.0480  0.8094771  0.5928616
##   0.0481  0.8094771  0.5928616
##   0.0482  0.8094771  0.5928616
##   0.0483  0.8094771  0.5928616
##   0.0484  0.8094771  0.5928616
##   0.0485  0.8094771  0.5928616
##   0.0486  0.8094771  0.5928616
##   0.0487  0.8094771  0.5928616
##   0.0488  0.8094771  0.5928616
##   0.0489  0.8094771  0.5928616
##   0.0490  0.8094771  0.5928616
##   0.0491  0.8094771  0.5928616
##   0.0492  0.8094771  0.5928616
##   0.0493  0.8050327  0.5844598
##   0.0494  0.8050327  0.5844598
##   0.0495  0.8050327  0.5844598
##   0.0496  0.8050327  0.5844598
##   0.0497  0.8050327  0.5844598
##   0.0498  0.8050327  0.5844598
##   0.0499  0.8050327  0.5844598
##   0.0500  0.8050327  0.5844598
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.0148.
```

Now I will build my CART model based on cp = 0.0149 and have a look on it

```
bestTree = rpart(Survived ~ Sex +
                 Pclass +
                 Embarked +
                 SibSp +
                 mother +
                 child +
                 Title +
                 familySize, data = train, method="class", cp = 0.0149)

prp(bestTree)
```



Surpisingly our classification tree looks quite simple.

Let's submit this version and see how it will get scored in Kaggle.

```
prediction3 <- predict(bestTree, newdata=test, type = "class")
solution3 <- data.frame(PassengerID = test$PassengerId, Survived = as.numeric(prediction3)-1)
write.csv(solution3, file = 'model3_Solution.csv', row.names = F)
```

This prediction scored 78.947% at Kaggle, which is a bit of disappointment. It seems that I am doing a principal error - in other words I need to take a second look at the data - manually and decide how to proceed.

Second round of Exploratory Analysis

```r
summary(train)
```

```
##   PassengerId    Survived Pclass      Name                Sex
##  Min.   :  1.0   0:549    1:216   Length:891         female:314
##  1st Qu.:223.5   1:342    2:184   Class :character   male  :577
##  Median :446.0            3:491   Mode  :character
##  Mean   :446.0
##  3rd Qu.:668.5
##  Max.   :891.0
##       Age            SibSp           Parch           Ticket
##  Min.   : 0.42   Min.   :0.000   Min.   :0.0000   Length:891
##  1st Qu.:20.00   1st Qu.:0.000   1st Qu.:0.0000   Class :character
##  Median :28.00   Median :0.000   Median :0.0000   Mode  :character
##  Mean   :29.53   Mean   :0.523   Mean   :0.3816
##  3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##  Max.   :80.00   Max.   :8.000   Max.   :6.0000
##       Fare           Cabin           Embarked   child
##  Min.   :  0.00   Length:891          :  2    Mode :logical
##  1st Qu.:  7.91   Class :character   C:168    FALSE:746
##  Median : 14.45   Mode  :character   Q: 77    TRUE :145
##  Mean   : 32.20                      S:644
##  3rd Qu.: 31.00
##  Max.   :512.33
##    familySize        Title              mother
##  Min.   : 1.000   Length:891         Mode :logical
##  1st Qu.: 1.000   Class :character   FALSE:835
##  Median : 1.000   Mode  :character   TRUE :56
##  Mean   : 1.905
##  3rd Qu.: 2.000
##  Max.   :11.000
```

```r
table(train$mother, train$Survived)/nrow(train)
```

```
##
##                  0          1
##   FALSE 0.59820426 0.33894501
##   TRUE  0.01795735 0.04489338
```

```r
table(train$Pclass, train$Survived)/nrow(train)
```

```
##
##              0          1
##   1 0.08978676 0.15263749
##   2 0.10886644 0.09764310
##   3 0.41750842 0.13355780
```

```r
table(train$familySize, train$Survived)/nrow(train)
```

```
##
##                0           1
##   1  0.419753086 0.182940516
##   2  0.080808081 0.099887767
##   3  0.048260382 0.066217733
##   4  0.008978676 0.023569024
```

```
##   5  0.013468013 0.003367003
##   6  0.021324355 0.003367003
##   7  0.008978676 0.004489338
##   8  0.006734007 0.000000000
##   11 0.007856341 0.000000000
```

```r
table(train$Title, train$Survived)/nrow(train)
```

```
##
##                          0           1
##   Master     0.019079686 0.025813692
##   Miss       0.061728395 0.145903479
##   Mr         0.489337823 0.090909091
##   Mrs        0.029180696 0.112233446
##   Rare Title 0.016835017 0.008978676
```

```r
table(train$Embarked, train$Survived)/nrow(train)
```

```
##
##               0           1
##     0.000000000 0.002244669
##   C 0.084175084 0.104377104
##   Q 0.052749719 0.033670034
##   S 0.479236813 0.243546577
```

```r
table(train$child, train$Survived)/nrow(train)
```

```
##
##                  0          1
##   FALSE 0.53759820 0.29966330
##   TRUE  0.07856341 0.08417508
```