

# Titanic

*Tianyuan Liu, Bochen Wang*

*September 2, 2015*

## Introduction

This is the second part of programming assignment 1.

## Data Cleaning

Load required R libraries and read data

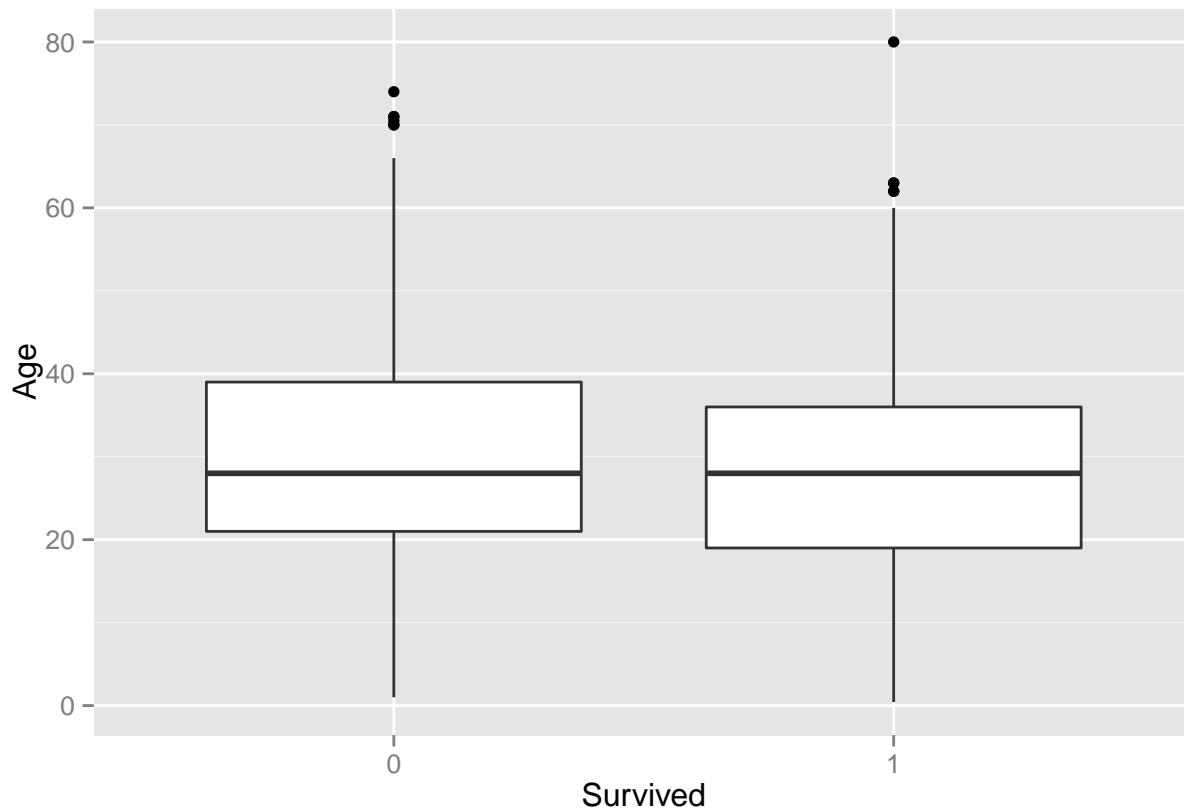
```
require(plyr)
require(ggplot2)
set.seed(35)
readData <- function(fileName) {
  data <- read.csv(file = fileName, header = T, na.strings=c("", "NA"))
  data$PassengerId <- as.integer(data$PassengerId)
  data$Pclass <- as.factor(data$Pclass)
  data$Ticket <- as.character(data$Ticket)
  data$Name <- as.character(data$Name)
  data$Cabin <- as.character(data$Cabin)
  data
}
train <- readData(fileName = "./train.csv")
train$Survived <- as.factor(train$Survived)
summary(train)
```

```
## PassengerId Survived Pclass      Name      Sex
## Min.      : 1      0:549      1:216  Length:891    female:314
## 1st Qu.:224      1:342      2:184  Class :character  male :577
## Median :446                      3:491  Mode  :character
## Mean      :446
## 3rd Qu.:668
## Max.      :891
##
##      Age      SibSp      Parch      Ticket
## Min.    : 0.42  Min.    :0.000  Min.    :0.000  Length:891
## 1st Qu.:20.12  1st Qu.:0.000  1st Qu.:0.000  Class :character
## Median :28.00  Median :0.000  Median :0.000  Mode  :character
## Mean     :29.70  Mean     :0.523  Mean     :0.382
## 3rd Qu.:38.00  3rd Qu.:1.000  3rd Qu.:0.000
## Max.     :80.00  Max.     :8.000  Max.     :6.000
## NA's     :177
##      Fare      Cabin      Embarked
## Min.    : 0.0  Length:891  C      :168
## 1st Qu.: 7.9  Class :character  Q      : 77
## Median :14.5  Mode  :character  S      :644
## Mean     :32.2  NA's: 2
```

```
## 3rd Qu.: 31.0
## Max.    :512.3
##
```

## Data Visualization and Feature Engineering

### Age and Survived rate

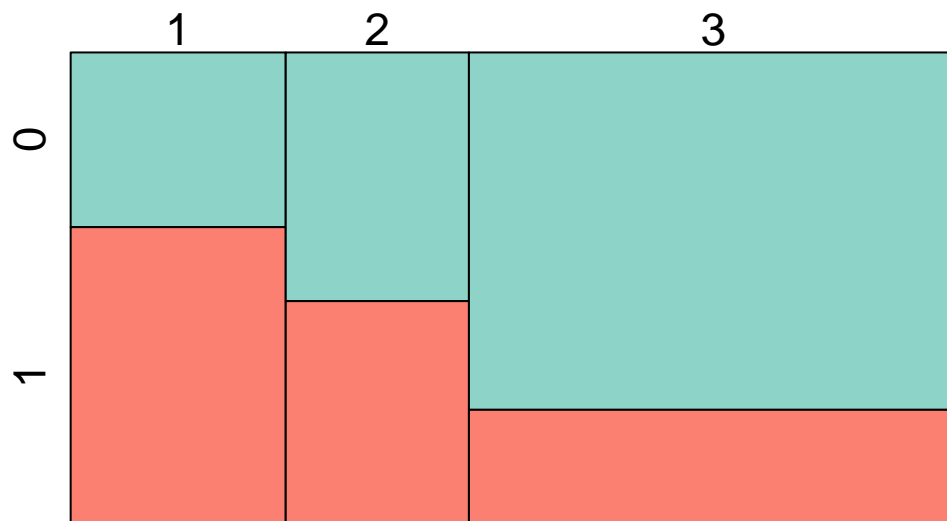


### Pclass is a good indicator

Using the plot method suggested by this [article](#), I managed to plot the relationship between survived and Pclass.

```
mosaicplot(train$Pclass ~ train$Survived, main="Passenger Survival by Class",
            color=c("#8dd3c7", "#fb8072"), shade=FALSE, xlab="", ylab="",
            off=c(0), cex.axis=1.4)
```

## Passenger Survival by Class



### Ignore Cabin feature

```
require(Hmisc)
describe(train$Cabin)
```

```
## train$Cabin
##      n missing  unique
##    204      687    147
##
## lowest : A10 A14 A16 A19 A20, highest: F33 F38 F4  G6  T
```

The missing data in **Cabin** is too much to interpolate, thus we will not use it as a feature in training model.

### Extracting titles from Name

```
head(train$Name)
```

```
## [1] "Braund, Mr. Owen Harris"
## [2] "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
## [3] "Heikkinen, Miss. Laina"
## [4] "Futrelle, Mrs. Jacques Heath (Lily May Peel)"
## [5] "Allen, Mr. William Henry"
## [6] "Moran, Mr. James"
```

According to [one tutorial](#) on this challenge, the [Wikipedia entry](#) for the [English honorific](#) “Master” explains that,

By the late 19th century, etiquette dictated that men be addressed as Mister, and boys as Master.

So I extracted the title feature from **Name**. And group ‘same’ titles together according to the rules suggested by [this article](#)

```
extractTitle <- function(data) {
  title <- sapply(data$Name, FUN = function(x) {strsplit(x, split="[,.]")[[1]][2]
})
  title <- gsub("^\\s+|\\s+$", "", title)
  title
}
groupTitle <- function(data, oldTitles, newTitle) {
  for (title in oldTitles) {
    data[data$Title == title, "Title"] <- newTitle
  }
  data$Title
}

train$Title <- extractTitle(train)
train$Title <- groupTitle(train, c("Capt", "Col", "Don",
                                   "Dr", "Jonkheer", "Lady",
                                   "Major", "Rev", "Sir"), "Upper Class")
train$Title <- groupTitle(train, c("the Countess", "Ms"), "Mrs")
train$Title <- groupTitle(train, c("Mlle", "Mme"), "Miss")
train$Title <- as.factor(train$Title)
```

### Impute null values in Age field and Embarked Field

```
summary(train$Age)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
##	0.42	20.10	28.00	29.70	38.00	80.00	177

If we look at the **Age** field, there are a lot of missing values. I interpolate on a **per-title** basis, and used **impute** method from **Hmisc** package.

```
imputeAge <- function(data) {
  age <- data$Age
  for (title in unique(data$Title)) {
    age[data$Title == title] <- impute(age[data$Title == title], fun = median)
  }
  age
}
train$Age <- imputeAge(train)
```

There is only two field missing in **Embarked**, I impute them with ‘S’, since ‘S’ is the most common element.

```
train[is.na(train$Embarked),]$Embarked <- 'S'
```

## Women and children first policy

Also I added a new feature indicating whether a person is a female or under age 12.

```
extractPolicy <- function(data) {  
  plc <- rep(0, length(data$Sex))  
  plc[data$Sex == 'female' | data$Age < 12] <- 1  
  factor(plc)  
}  
train$Plc <- extractPolicy(train)  
chisq.test(table(train$Plc, train$Survived))
```

```
##  
## Pearson's Chi-squared test with Yates' continuity correction  
##  
## data:  table(train$Plc, train$Survived)  
## X-squared = 278.9, df = 1, p-value < 2.2e-16
```

I conducted the chi-square test on the new feature, since p-value is smaller than **.05**, we can reject the null hypothesis that the policy feature is independent of the survive rate.

## Family feature

I added **SibSp** and **Parch** together to form a new feature indicating the number of family members.

```
extractFamily <- function(data) {  
  data$SibSp + data$Parch  
}  
train$Family <- extractFamily(train)
```

## Model Fitting and tuning

### fit model

First, I will use all the variable(except **SibSp** and **Parch**, instead I used the **Family** feature to represent) to train a logistic regression model.

```
train.model <- glm(Survived ~ Pclass + Sex + Age + Title +  
                  Plc + Fare + Embarked + Family, data = train,  
                  family = binomial(link=logit))  
summary(train.model)
```

```
##  
## Call:  
## glm(formula = Survived ~ Pclass + Sex + Age + Title + Plc + Fare +  
##      Embarked + Family, family = binomial(link = logit), data = train)  
##
```

```
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.365   -0.565   -0.381    0.551    2.516
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    4.59e+01  1.38e+03   0.03  0.97353
## Pclass2       -1.07e+00  3.22e-01  -3.32  0.00089 ***
## Pclass3       -2.16e+00  3.19e-01  -6.78  1.2e-11 ***
## Sexmale       -2.85e+01  8.78e+02  -0.03  0.97406
## Age          -2.91e-02  9.62e-03  -3.02  0.00249 **
## TitleMiss     -2.89e+01  8.78e+02  -0.03  0.97371
## TitleMr       -1.65e+01  6.17e+02  -0.03  0.97874
## TitleMrs      -2.80e+01  8.78e+02  -0.03  0.97454
## TitleUpper Class -1.64e+01  6.17e+02  -0.03  0.97876
## Plc1          -1.32e+01  6.17e+02  -0.02  0.98299
## Fare           3.61e-03  2.66e-03   1.36  0.17358
## EmbarkedQ     -1.59e-01  3.96e-01  -0.40  0.68829
## EmbarkedS     -4.31e-01  2.52e-01  -1.71  0.08662 .
## Family        -4.62e-01  8.45e-02  -5.46  4.7e-08 ***
## ---
## 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:  724.34  on 877  degrees of freedom
## AIC: 752.3
##
## Number of Fisher Scoring iterations: 13
```

From the summary we can see, only a few of variables has passed the **z-test**, which are **Pclass**, **SibSp** and **Parch**. For others, we can not reject the null hypothesis that the corresponding coefficient being zero.

## Model Selection based on AIC

So next I conducted feature selection based on AIC.

```
step(train.model, trace = 0)
```

```
##
## Call:  glm(formula = Survived ~ Pclass + Sex + Age + Title + Fare +
##      Family, family = binomial(link = logit), data = train)
##
## Coefficients:
##      (Intercept)      Pclass2      Pclass3      Sexmale
##      19.39894      -1.16663      -2.17861     -15.42092
##           Age      TitleMiss      TitleMr      TitleMrs
##      -0.02977     -15.83491     -3.36616     -14.93213
## TitleUpper Class      Fare      Family
##      -3.27790      0.00434     -0.48189
##
## Degrees of Freedom: 890 Total (i.e. Null);  880 Residual
```

```
## Null Deviance:      1190
## Residual Deviance: 729   AIC: 751
```

Ok, so I trained next model based on AIC suggestion.

```
train.model <- glm(formula = Survived ~ Pclass + Sex + Age +
                    Title + Fare + Family, family = binomial(link = logit),
                    data = train)
summary(train.model)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age + Title + Fare +
##      Family, family = binomial(link = logit), data = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.444  -0.555  -0.389   0.541   2.592
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    19.39894   622.00349     0.03  0.97512
## Pclass2        -1.16663     0.31795    -3.67  0.00024 ***
## Pclass3        -2.17861     0.31173    -6.99  2.8e-12 ***
## Sexmale       -15.42092   622.00323    -0.02  0.98022
## Age            -0.02977     0.00957    -3.11  0.00186 **
## TitleMiss     -15.83491   622.00342    -0.03  0.97969
## TitleMr        -3.36616     0.52734    -6.38  1.7e-10 ***
## TitleMrs     -14.93213   622.00347    -0.02  0.98085
## TitleUpper Class -3.27790     0.77688    -4.22  2.5e-05 ***
## Fare           0.00434     0.00264     1.64  0.10001
## Family        -0.48189     0.08317    -5.79  6.9e-09 ***
## ---
## 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:  728.51  on 880  degrees of freedom
## AIC: 750.5
##
## Number of Fisher Scoring iterations: 13
```

From the summary we can see, Title **Mr** and **Upper Class** seem to be a good indicator, but other titles are not. So I am going to drop **Miss** and **Mrs**. Also, **fare** and **Sex** can be dropped.

```
train.model <- glm(formula = Survived ~ Pclass + Age +
                    I(Title=="Mr") + I(Title=="Upper Class") +
                    Family, family = binomial(link = logit),
                    data = train)
summary(train.model)
```

```
##
```

```
## Call:
## glm(formula = Survived ~ Pclass + Age + I(Title == "Mr") + I(Title ==
##      "Upper Class") + Family, family = binomial(link = logit),
##      data = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.402  -0.591  -0.389   0.561   2.574
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)      3.88201    0.40082    9.69 < 2e-16 ***
## Pclass2         -1.29825    0.27466   -4.73 2.3e-06 ***
## Pclass3         -2.41402    0.25871   -9.33 < 2e-16 ***
## Age             -0.02111    0.00799   -2.64 0.0083 **
## I(Title == "Mr")TRUE -3.31258    0.22748  -14.56 < 2e-16 ***
## I(Title == "Upper Class")TRUE -3.08798    0.53498   -5.77 7.8e-09 ***
## Family          -0.39884    0.07065   -5.65 1.6e-08 ***
## ---
## 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:  742.66  on 884  degrees of freedom
## AIC: 756.7
##
## Number of Fisher Scoring iterations: 5
```

## Confusion Matrix

Finally I build a confusion matrix.

```
Survived <- predict(train.model, newdata = train, type="response")
predictions <- as.data.frame(Survived)
predictions$Survived <- sapply(predictions$Survived,
                                function(x) {ifelse(x >= 0.5,
                                                        "Predicted Survived",
                                                        "Predicted Not Survived")})
predictions$Truth <- train$Survived
prop.table(table(predictions$Truth, predictions$Survived), 2)
```

```
##
##      Predicted Not Survived Predicted Survived
##      0              0.8387              0.1821
##      1              0.1613              0.8179
```

## Submit Result

The following script does the same processing procedures on test dataset, and save the prediction to csv file.





```

test <- readData('./test.csv')
test$Title <- extractTitle(test)
test$Title <- groupTitle(test, c("Capt", "Col", "Don",
                                "Dr", "Jonkheer", "Lady",
                                "Major", "Rev", "Sir"), "Upper Class")
test$Title <- groupTitle(test, c("the Countess", "Ms"), "Mrs")
test$Title <- groupTitle(test, c("Mlle", "Mme"), "Miss")
test$Age <- imputeAge(test)
test$Family <- extractFamily(test)
# use probability 0.5 as a threshold.
Survived <- predict(train.model, newdata = test, type="response")
predictions <- as.data.frame(Survived)
predictions$PassengerId <- test[,1]
predictions$Survived <- sapply(predictions$Survived, function(x) {ifelse(x >= 0.5, 1, 0)})

write.csv(predictions, file='result.csv', row.names=F, quote=F)

```

I achieved **0.78947** accuracy rate on kaggle's data.

1303		AndyLiu	0.78947	5	Sun, 06 Sep 2015 22:13:46 (-0.6h)
1304		linalg	0.78947	2	Mon, 07 Sep 2015 18:40:03