Titanic Data Analysis

Yakub Akhmerov

November 29, 2016

Intro

After spending some time searching through the datasets on Kaggle.com, I came across the titanic dataset. I chose this because not only was it a significant historical event globally, its history was also nodable in the technilogical aspect of the world. Since my interest lie in technology, this analysis would be a good introduction in truly applied data analytics. What follows is the report which was used for this analysis.

Goal

The aim was to use the training dataset to predict the survival status of those in the testing dataset. I chose to do this by the approach of fitting a generalized linear model where survival status is the binary reponse variable (logistic regression). For the purposes of clarity, I will document my steps at the approrpriate coding chunks in the report. I start off by choosing the variables which are of revelance to the desired outcome. For example, ticket fare shouldn't have much of an effect on whether someone survived or not. While the number of siblings a person has will.

Load data & packages

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(SDMTools)
setwd("~/Downloads")
test <- read.csv('test.csv') #read in test data
train <- read.csv('train.csv') #read in train data
train <- train %>%
  mutate(Sex = ifelse(Sex=="male", 1, 0)) %>%
  mutate(Embarked = ifelse(Embarked=="C", 1, ifelse(Embarked=="S", 2, 3)))
#Creating a quantitative representation of the Sex and Embarked by wrangling the data
#as there are at most 3 different categories between the 2 variables.
```

```
#I examined the pairs of the functions and for the purposes of modeling, I found it
#necessary to exclude the following variables: PassengerId, Name, Age, Ticket, Fare, and Cabin, as
#all of them had several different variation of values which were not relevent to the purposes
#of our study. I then proceeded to fit the model with the remaining variables.

titanic.glm <- glm(Survived ~ Sex + Pclass + SibSp + Parch + Embarked, family=binomial, data=train)
```

Fitting best model.

I the performed model diagnostics in order to choose the best fit model.

```
names(titanic.glm)
```

```
##  [1] "coefficients"      "residuals"         "fitted.values"
##  [4] "effects"           "R"                 "rank"
##  [7] "qr"                "family"            "linear.predictors"
## [10] "deviance"          "aic"               "null.deviance"
## [13] "iter"              "weights"           "prior.weights"
## [16] "df.residual"       "df.null"           "y"
## [19] "converged"         "boundary"          "model"
## [22] "call"              "formula"           "terms"
## [25] "data"              "offset"            "control"
## [28] "method"            "contrasts"         "xlevels"
```

```
titanic.glm$coef
```

```
## (Intercept)         Sex      Pclass       SibSp       Parch    Embarked
##  3.83333994 -2.78454290 -0.89918285 -0.23574909 -0.06339667 -0.23367940
```

```
mean(titanic.glm$fitted.values) #estimated
```

```
## [1] 0.3838384
```

```
#Was curious if any variables neede to be thrown out. I conducted AIC and BIC to determine this.
step(titanic.glm) #AIC
```

```
## Start:  AIC=829.33
## Survived ~ Sex + Pclass + SibSp + Parch + Embarked
##
##            Df Deviance     AIC
## - Parch     1   817.66  827.66
## - Embarked  1   819.11  829.11
## <none>          817.33  829.33
## - SibSp     1   823.57  833.57
## - Pclass    1   888.89  898.89
## - Sex       1  1074.42 1084.42
##
## Step:  AIC=827.66
## Survived ~ Sex + Pclass + SibSp + Embarked
```

```
##
##             Df Deviance     AIC
## - Embarked  1   819.32  827.32
## <none>          817.66  827.66
## - SibSp     1   825.79  833.79
## - Pclass    1   889.45  897.45
## - Sex       1  1083.95 1091.95
##
## Step:  AIC=827.32
## Survived ~ Sex + Pclass + SibSp
##
##          Df Deviance     AIC
## <none>       819.32  827.32
## - SibSp   1   827.20  833.20
## - Pclass  1   904.69  910.69
## - Sex     1  1084.34 1090.34
```

```
##
## Call:  glm(formula = Survived ~ Sex + Pclass + SibSp, family = binomial,
##     data = train)
##
## Coefficients:
## (Intercept)          Sex       Pclass        SibSp
##      3.4336      -2.7431      -0.9390      -0.2481
##
## Degrees of Freedom: 890 Total (i.e. Null);  887 Residual
## Null Deviance:       1187
## Residual Deviance: 819.3     AIC: 827.3
```

```
step(titanic.glm, direction="both", k = log(nrow(train))) #BIC
```

```
## Start:  AIC=858.09
## Survived ~ Sex + Pclass + SibSp + Parch + Embarked
##
##             Df Deviance     AIC
## - Parch     1   817.66  851.62
## - Embarked  1   819.11  853.07
## - SibSp     1   823.57  857.54
## <none>          817.33  858.09
## - Pclass    1   888.89  922.85
## - Sex       1  1074.42 1108.39
##
## Step:  AIC=851.62
## Survived ~ Sex + Pclass + SibSp + Embarked
##
##             Df Deviance     AIC
## - Embarked  1   819.32  846.49
## <none>          817.66  851.62
## - SibSp     1   825.79  852.96
## + Parch     1   817.33  858.09
## - Pclass    1   889.45  916.62
## - Sex       1  1083.95 1111.12
##
```

```
## Step:  AIC=846.49
## Survived ~ Sex + Pclass + SibSp
##
##             Df Deviance     AIC
## <none>           819.32  846.49
## - SibSp      1   827.20  847.57
## + Embarked   1   817.66  851.62
## + Parch      1   819.11  853.07
## - Pclass     1   904.69  925.07
## - Sex        1  1084.34 1104.72

##
## Call:  glm(formula = Survived ~ Sex + Pclass + SibSp, family = binomial,
##     data = train)
##
## Coefficients:
## (Intercept)          Sex       Pclass        SibSp
##      3.4336      -2.7431      -0.9390      -0.2481
##
## Degrees of Freedom: 890 Total (i.e. Null);  887 Residual
## Null Deviance:       1187
## Residual Deviance: 819.3     AIC: 827.3
```

```
#Both the AIC and BIC had Survived ~ Sex + Pclass + SibSp as the final model, both
#resulting with a residual deviance of 819.3

titanic.glm <- glm(Survived ~ Sex + Pclass + SibSp, family=binomial, data=train)
```

Finding threshold

In prediction analysis, it is best to pick a threshold which maximizes accuracy. Accuracy being the number of (TP + TN)/(P+N). Where TP is the number of True Positives (predicted to be positive and actually positive), TN is the number of True Negative(predicted negative, actually negative), and P and N being the number of real positives and real negatives respectively.

```
#I discovered the confusion.matrix function through the SDMTools Rpackage, which computes
#the True Positives and Negatives along with the False Positives and Negatives
#In order to choose the model with the best threshold. I generated 21 confusion matrices
#with thresholds ranging from .25 to .975 that I choose and selected the threshold with
#the most highest accuracy
n=nrow(train)
m0 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .025)
m1 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .5)
m2 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .10)
m3 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .15)
m4 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .20)
m5 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .25)
m6 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .30)
m7 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .35)
m8 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .40)
m9 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .45)
m10 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .50)
```

```
m11 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .55)
m12 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .60)
m13 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .65)
m14 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .70)
m15 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .75)
m16 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .80)
m17 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .85)
m18 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .90)
m19 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .95)
m20 = confusion.matrix(obs = titanic.glm$y, titanic.glm$fitted.values, threshold = .975)

max( (m0[1,1] + m0[2,2]) / n, (m1[1,1] + m1[2,2])/n, (m2[1,1] + m2[2,2])/n,
     (m3[1,1] + m3[2,2])/n, (m4[1,1] + m4[2,2])/n, (m5[1,1] + m5[2,2])/n,
     (m6[1,1] + m6[2,2])/n, (m7[1,1] + m7[2,2])/n, (m8[1,1] + m8[2,2])/n,
     (m9[1,1] + m9[2,2])/n, (m10[1,1] + m10[2,2])/n, (m11[1,1] + m11[2,2])/n,
     (m12[1,1] + m12[2,2])/n, (m13[1,1] + m13[2,2])/n, (m14[1,1] + m14[2,2])/n,
     (m15[1,1] + m15[2,2])/n, (m16[1,1] + m16[2,2])/n, (m17[1,1] + m17[2,2])/n,
     (m18[1,1] + m18[2,2])/n, (m19[1,1] + m19[2,2])/n, (m20[1,1] + m20[2,2])/n)
```

```
## [1] 0.8035915
```

```
#The highest accuracy was 0.8035915, which was m12 with a threshold of .60

(m12[1,1] + m12[2,2])/n
```

```
## [1] 0.8035915
```

Granted, this approach was rather brute forced. This is due to my mathematical background in linear algebra, abstract algebra, and linear regression. I understood this theory behind the confusion matrix and accuracy through this approach. As an individual with such a background, the concreteness was enough for me to stick with it.

Applying the new knowledge to our train dataset

```
#The purpose of yhat was to create a function to check which values pass the threshold
#in our model. I then planned to store them in a vector.
yhat <- function(c, phat){ifelse(phat > c, 1, 0)}

test <- test %>%
  mutate(Sex = ifelse(Sex=="Male", 1, 0)) %>%
  mutate(Embarked = ifelse(Embarked=="C", 1, ifelse(Embarked=="S", 2, 3)))
#replicate the same quantitative variables as our model

prediction <- predict(titanic.glm, test, type = "response")

phats <- yhat(.58, predict(titanic.glm, test, type = "response"))

solution <- data.frame(PassengerID = test$PassengerId, Survived = phats)

write.csv(solution, file = "solution5.csv", row.names=F)
```