# SML - Assignment 2

Harshad Kumar Elangovan - 19200349

02/04/2020

The dataset “spam” is an extract of the contents of emails which were classified as spam and non spam emails. It has a total of 58 columns in which first 48 contain the frequency of the variable name in the email.The columns 49-54 indicate the frequency of the special characters & 55-57 contain the average, longest and total run-length of capital letters. The last column “type” indicates if an email is spam or nonspam.This last column “type” is the target response variable while fitting the Logistic Regression model to the data.

```
library(kernlab)
data(spam)
emaildata<-data.frame(spam)
```

This emaildata data frame contains all the variables of the spam dataset. For fitting the regression, we have to subset the dataset based on the frequencies of special characters. So, emaildata is updated and the variable “type” is updated from “spam” or “nonspam” to 0 or 1 for further usage of the dataset in the model.

```
emaildata1<- emaildata[,49:58]

#Converting the factors to 0's and 1's
library(dplyr)
emaildata1$type<-recode(emaildata1$type,'spam'=1,'nonspam'=0)
str(emaildata1)
```

```
## 'data.frame':    4601 obs. of  10 variables:
##  $ charSemicolon    : num  0 0 0.01 0 0 0 0 0 0 0.04 ...
##  $ charRoundbracket : num  0 0.132 0.143 0.137 0.135 0.223 0.054 0.206 0.271 0.0
3 ...
##  $ charSquarebracket: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ charExclamation  : num  0.778 0.372 0.276 0.137 0.135 0 0.164 0 0.181 0.244
...
##  $ charDollar       : num  0 0.18 0.184 0 0 0 0.054 0 0.203 0.081 ...
##  $ charHash         : num  0 0.048 0.01 0 0 0 0 0 0.022 0 ...
##  $ capitalAve       : num  3.76 5.11 9.82 3.54 3.54 ...
##  $ capitalLong      : num  61 101 485 40 40 15 4 11 445 43 ...
##  $ capitalTotal     : num  278 1028 2259 191 191 ...
##  $ type             : num  1 1 1 1 1 1 1 1 1 1 ...
```

## Fitting the Logistic Regression Model

The updated dataset emaildata1 is then used for fitting the model. This can be done using glm command.

```
#Fitting the model
fit<-glm(emaildata1$type ~.,data = emaildata1,family = "binomial")
summary(fit)
```

```
##
## Call:
## glm(formula = emaildata1$type ~ ., family = "binomial", data = emaildata1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -8.4904  -0.6403  -0.5211   0.5177   3.6202
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.677e+00  7.038e-02 -23.835  < 2e-16 ***
## charSemicolon     -1.055e+00  4.117e-01  -2.562 0.010419 *
## charRoundbracket  -1.441e+00  2.513e-01  -5.733 9.87e-09 ***
## charSquarebracket -3.878e+00  1.085e+00  -3.574 0.000351 ***
## charExclamation    1.312e+00  1.100e-01  11.931  < 2e-16 ***
## charDollar         1.059e+01  6.007e-01  17.622  < 2e-16 ***
## charHash           3.553e-01  1.445e-01   2.459 0.013924 *
## capitalAve         5.560e-02  2.195e-02   2.533 0.011308 *
## capitalLong        1.385e-02  1.653e-03   8.377  < 2e-16 ***
## capitalTotal       1.687e-04  8.902e-05   1.895 0.058034 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6170.2  on 4600  degrees of freedom
## Residual deviance: 4042.6  on 4591  degrees of freedom
## AIC: 4062.6
##
## Number of Fisher Scoring iterations: 8
```

**From the summary, we can see that the ‘charDollar’ & ‘charExclamation’ significantly affect the probability of an email being spam. These two variables are highly significant with p value less than 0.05**.

We will have to estimate the log-odds for checking the inferential problems related to these variables.

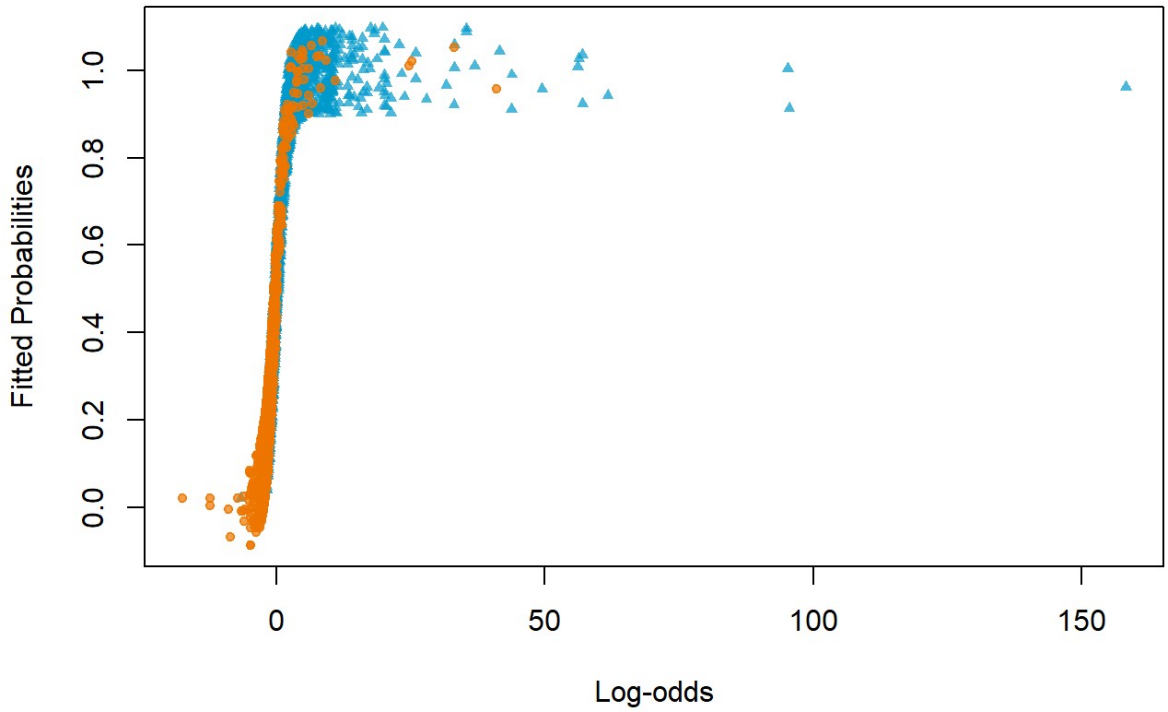
We can construct the Confidence Intervals for the estimated coefficients to verify the odds are in range and also check the confidence intervals of each coefficients.

```
#Confidence Intervals and log-odds
# Extracting the coefficients
w <-coef(fit)


#Computing 95% confidence interval of w and odds
#extracting Standard Errors
summaryfit<-summary(fit)
se<-summaryfit$coef[,2]

# compute confidence limits for w
wLowBound   <- w - 1.96 * se
wUpperBound <- w + 1.96 * se

# store coefficients and confidence limits
ci<- cbind(lb =wLowBound,w =w,ub =wUpperBound)
#ci

#confidence Intervals for odds
exp(ci)
```

```
##                              lb            w           ub
## (Intercept)       1.627849e-01 1.868614e-01 2.144989e-01
## charSemicolon     1.554098e-01 3.482977e-01 7.805900e-01
## charRoundbracket  1.446351e-01 2.367101e-01 3.874003e-01
## charSquarebracket 2.467565e-03 2.069274e-02 1.735271e-01
## charExclamation   2.994073e+00 3.714308e+00 4.607797e+00
## charDollar        1.219132e+04 3.957308e+04 1.284544e+05
## charHash          1.074795e+00 1.426638e+00 1.893660e+00
## capitalAve        1.012658e+00 1.057176e+00 1.103652e+00
## capitalLong       1.010664e+00 1.013944e+00 1.017234e+00
## capitalTotal      9.999943e-01 1.000169e+00 1.000343e+00
```

For estimating the log-odds, we will make use of ‘predict’ function in fit data. This estimate is then used for plotting the data.

```
#Estimated log-odds
lg<-predict(fit)
phat<-predict(fit,type="response")


#Plotting
symb<-c(19,17)
col<-c("darkorange2","deepskyblue3")
#jitter is used for adding noise for better visualization of the data.
plot(lg,jitter(phat,amount = 0.1),pch=symb[emaildata1$type +1],
     col = adjustcolor(col[emaildata1$type +1],0.7),cex = 0.7,
     xlab = "Log-odds",ylab="Fitted Probabilities")
```




There is a steep increase in the curve at x=0. This might be due to ‘w’ value close to infinity. his type of data lead to the problem of **complete separation** in logistic regression.This regresssion tries to fit a sigmoid curve to the data. But the coefficient ‘w’ is increased to infinity value and it becomes difficult to calculate the slope for the curve. This shows that **the model is fitted so good that it cannot be inferred with those variables**.

**Firth Logistic Regression or Regularized Logistic Regression** can be used in solving this inferential problem.

## Evaluate the general classification accuracy of the model using the ROC curve and calculate an optimal threshold τ for prediction.

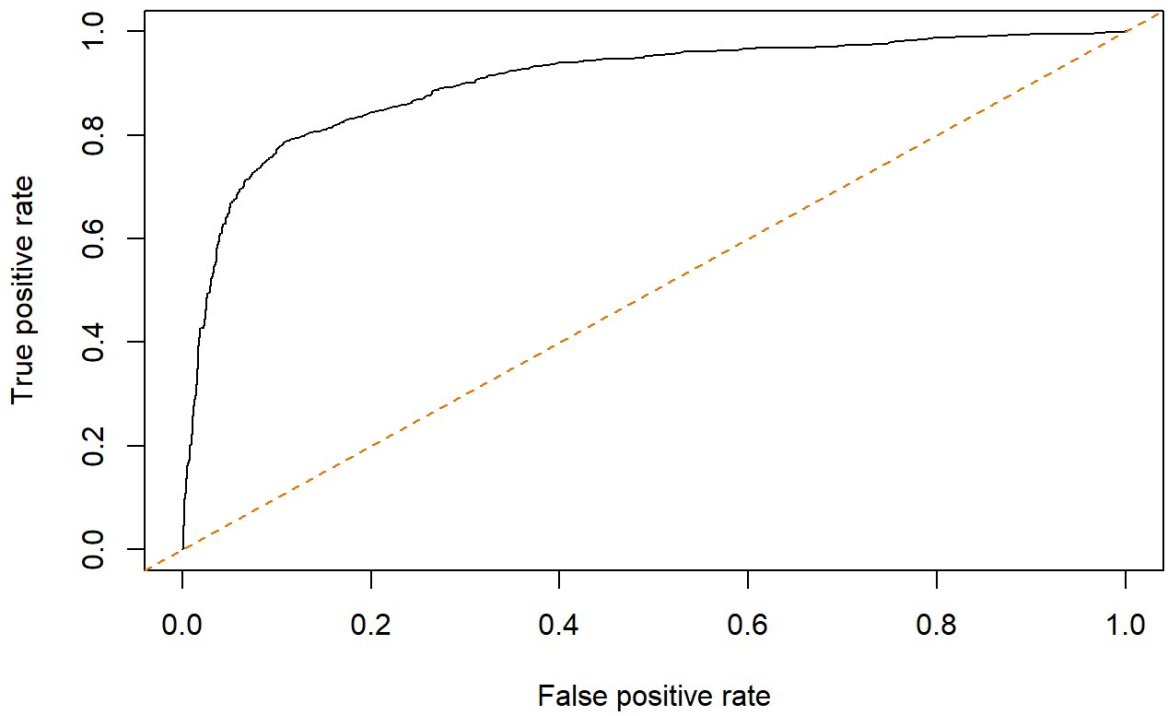
The Receiver Operating Characteristic(ROC) curve illustrates the diagnostic ability of a binary classifier as the discrimination threshold tau is varied. The estimated probabilities can be calculated uing a threshold tau. We set a default value of this threshold to 0.5.

```
#Evaluating the classification using a default threshold value
tau<-0.5
pred<-ifelse(fitted(fit)>tau,1,0)

#cross tabulation between observed and predicted
table(emaildata1$type,pred)
```

```
##    pred
##        0    1
##   0 2645  143
##   1  604 1209
```

we observe that 2645 records were correctly classified as nonspam and 1209 records were correctly identified as spam.Rest of the records are incorrectly classified. So we will use ROCR package for assessing the predictive performance for different values of discrimination threshold.

```
#ROCR
library(ROCR)

#Prediction object is created using prediction function
predObj<-prediction(fitted(fit),emaildata1$type)

#The performance measures are calculated using the prediction object with True and
False positive rate. This is then used for plotting the ROC curve.
perf<-performance(predObj,"tpr","fpr")
plot(perf)
abline(0,1,col="darkorange2",lty=2)
```

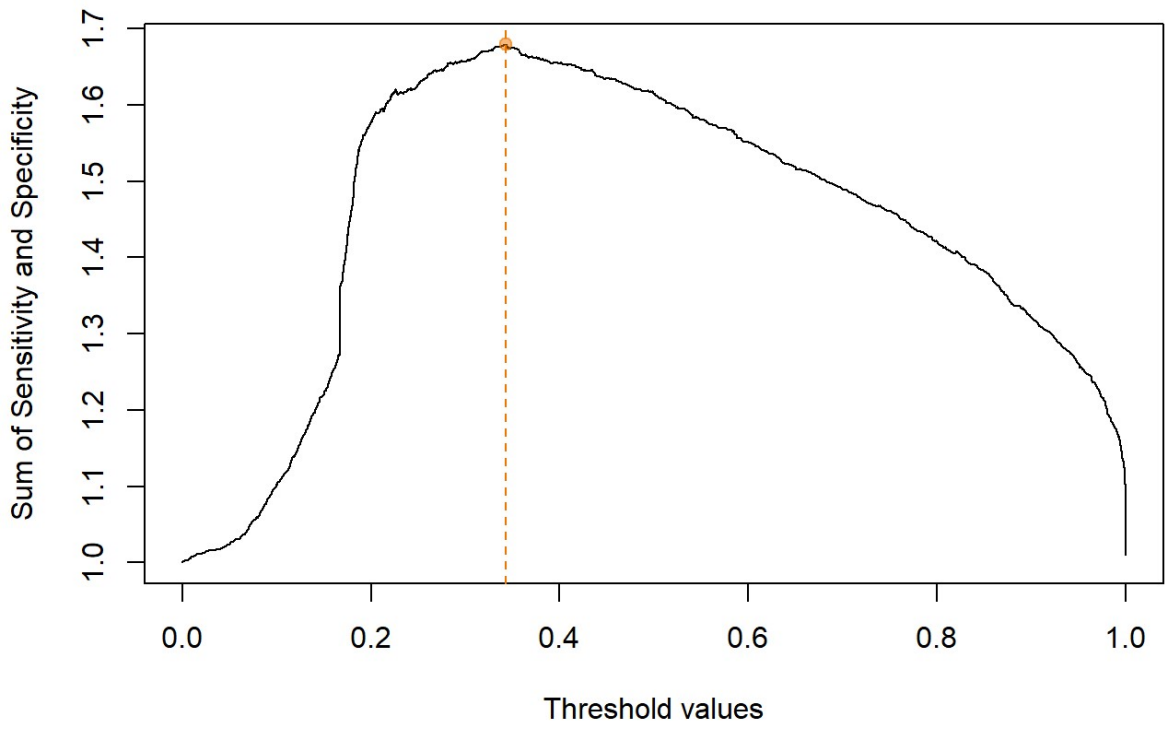



```
#The area under the curve is also calculated using the performance function.
auc<-performance(predObj,"auc")
auc@y.values
```

```
## [[1]]
## [1] 0.9020404
```

So,this classification of graph was done with a default threshold value 0.5. This classification can be improved using an optimal threshold value. This can be done by maximizing the sum of sensitivity and specificity for different threshold values.

```
#Sensitivity and Specificity calculation for optimal threshold value
sens<-performance(predObj,"sens")
spec<-performance(predObj,"spec")

tau<-sens@x.values[[1]]

#Sum of Sensitivity and Specificity
sensSpec<-sens@y.values[[1]] + spec@y.values[[1]]

#Finding the maximum of Sum of Sensitivity and Specificity
best<-which.max(sensSpec)

#
plot(tau,sensSpec,type="l",xlab="Threshold values",ylab="Sum of Sensitivity and Spe
cificity")
points(tau[best],sensSpec[best],pch=19,col=adjustcolor("darkorange2",0.5))
abline(v=tau[best],col="darkorange2",lty=2)
```




From this plot, we can find the optimal threshold value,pointed in orange color, based on the values of Sensitivity and Specificity.

The Optimal threshold and its classification is given by

```
tau[best]
```

```
##       195
## 0.3432798
```

```
#Classification for optimal threshold
pred<-ifelse(fitted(fit)>tau[best],1,0)
table(emaildata1$type,pred)
```

```
##    pred
##        0    1
##   0 2488  300
##   1  388 1425
```

From the above performance measures, the optimal threshold τ for prediction is calculated as 0.3432798 and from the Classification using this optimal threshold,we observe that 2488 records were correctly classified as nonspam and 1425 were correctly classified as spam. This optimal classification is better than the classification that was calculated with threshold value 0.5.