# Logistic Regression

*Justin Besteman*

*April 8, 2017*

## Clean up the past

```r
# Clean up
rm(list = ls())
```

## Loading Libraries

```r
library(readr)
library(ROCR)
```

## Reading in the data of PassFail.dat

```r
# Loading in Data
theData <- read_delim(
  "~/topics/hw4/PassFail.dat",
  " ",
  escape_double = FALSE,
  col_names = FALSE,
  col_types =
    cols(
      X1 = col_skip(),
      X10 = col_skip(),
      X12 = col_skip(),
      X14 = col_skip(),
      X2 = col_skip(),
      X4 = col_skip(),
      X6 = col_skip(),
      X8 = col_skip()
    ),
  na = "null",
  trim_ws = TRUE
)

# Making the theData into a data.frame
PassFail <- data.frame(theData)

# Renaming columns
colnames(PassFail) <- c("y", "x1", "x2", "x3", "x4", "x5", "x6")
```

## Number of Observations

```r
# Number of Observation will hold the number of observation
numberOfObservation <-  nrow(PassFail)

numberOfObservation
```

```
## [1] 10000
```

### Making sample data and test data

```r
set.seed(123321)

index <- c(1:numberOfObservation)

random6000 <- sample(index, numberOfObservation * .60)

trainingSamplePassFail <- PassFail[random6000 ,]

testData <-  PassFail[-random6000 ,]
```

### Running GLM

```r
model <- glm(y  ~ . , data = trainingSamplePassFail , family = binomial(link = "logit"))

model
```

```
##
## Call:  glm(formula = y ~ ., family = binomial(link = "logit"), data = trainingSamplePassFail)
##
## Coefficients:
## (Intercept)           x1           x2           x3           x4
##    -1.60329      1.57422      0.81495      0.40369      0.19858
##          x5           x6
##     0.07576      0.07100
##
## Degrees of Freedom: 5999 Total (i.e. Null);  5993 Residual
## Null Deviance:       8132
## Residual Deviance: 7608  AIC: 7622
```

### Observation scoring the test data

```r
fitted.results <- predict(model, testData ,type='response')
fitted.results <- ifelse(fitted.results > 0.5,1,0)

misClasificError <- mean(fitted.results != testData$y)
print(paste('Accuracy',1-misClasificError))
```

```
## [1] "Accuracy 0.65175"
```

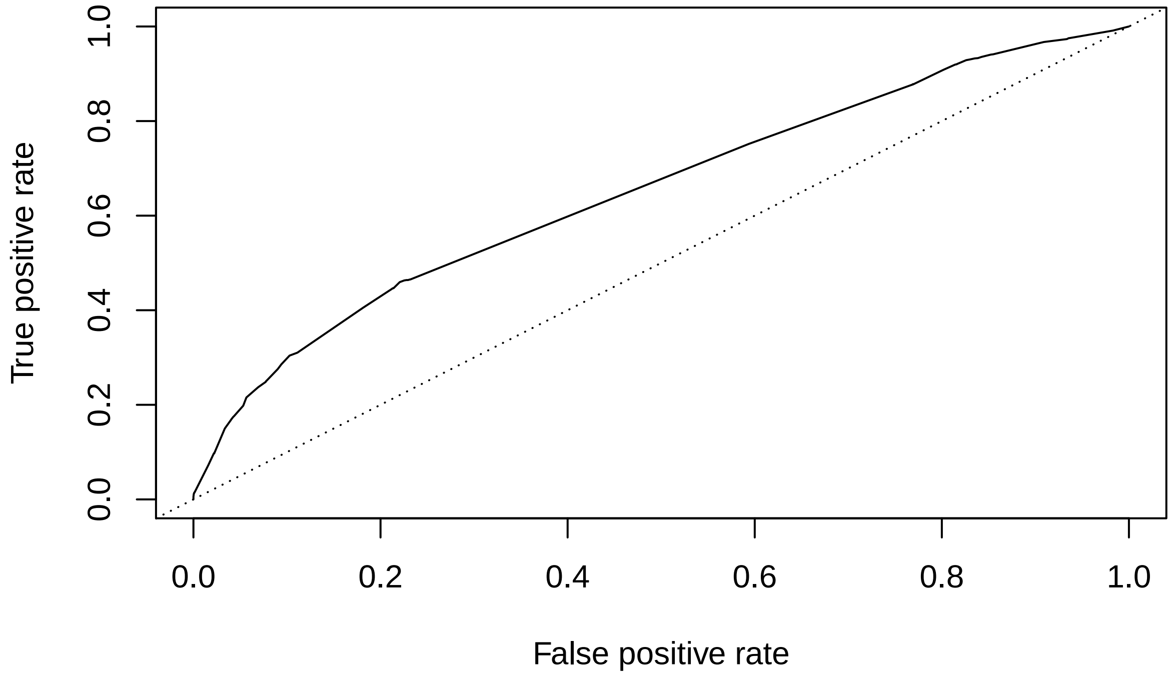
### Making the plot of the curve of the ROCR

```
p <- predict(model,testData, type="response")
pr <- prediction(p, testData$y)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")

auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.650838
```

```
plot(prf)
abline(a = 0 , b = 1 , lty = 3)
```


True positive rate
False positive rate