# Assignment 4

*Loic Roldan Waals*

*April 16, 2018*

## Question 2

First the data is imported

```
#determine where files are read and written
setwd("C:/Users/Loic RW/Google Drive/Big Data and Business Analytics/Assignments/Assignment 4")


#read all the data
library(readr)
tweets <- read_delim("C:/Users/Loic RW/Google Drive/Big Data and Business Analytics/Assignme
                     ";", escape_double = FALSE, col_types = cols(favorited = col_skip(),
                                                                  favoriteCount = col_skip(), isRetweet
                                                                  latitude = col_skip(), longitude = col
                                                                  replyToSID = col_skip(), replyToSN = 
                                                                  replyToUID = col_skip(), retweeted = 
                                                                  statusSource = col_skip(), truncated =

                     trim_ws = TRUE)
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)

## Warning: 4 parsing failures.
## row # A tibble: 4 x 5 col     row col   expected   actual     file
```

```
tweets$retweetCount <- as.numeric(tweets$retweetCount)
```

```
## Warning: NAs introduced by coercion
```

```
#delete 4 NA's for "retweetcount"
tweets <- tweets[complete.cases(tweets),]
```

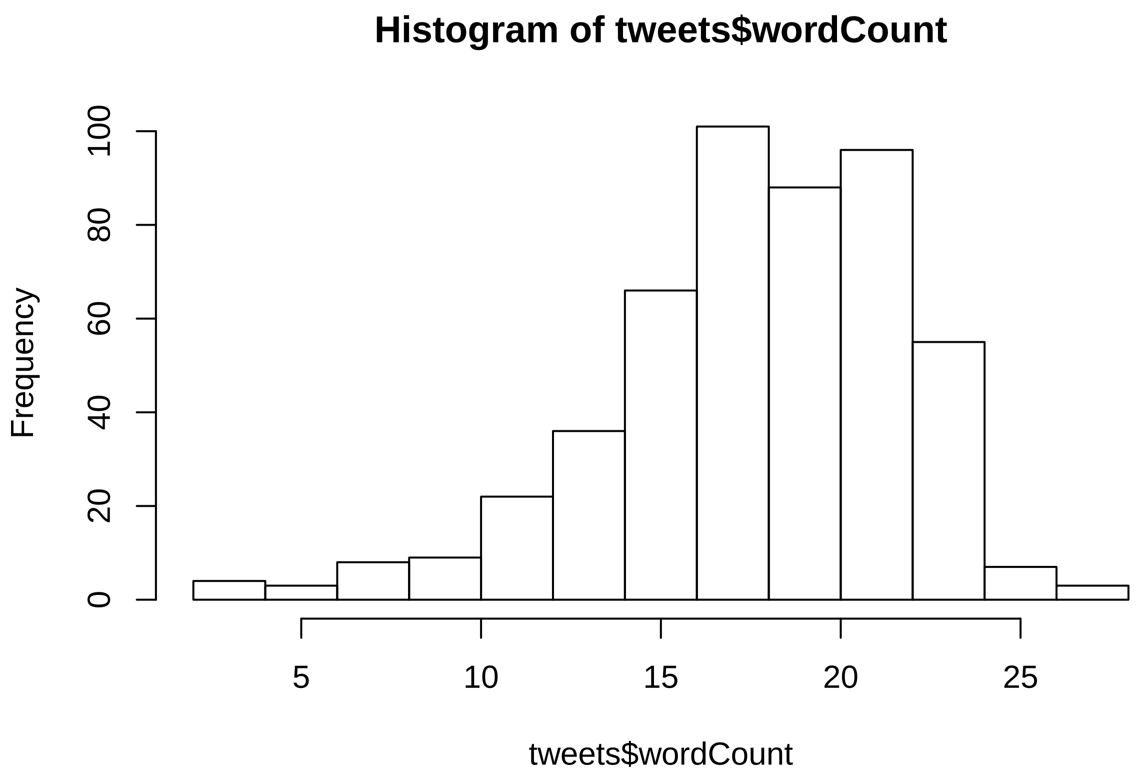
Next we make a histogram of the number of words of the tweets.

```
library(ngram)

for (i in 1:nrow(tweets)) {
  tweets$wordCount[i] <- as.numeric(wordcount(tweets$text[i], sep = " ", count.function = sum))
}
```

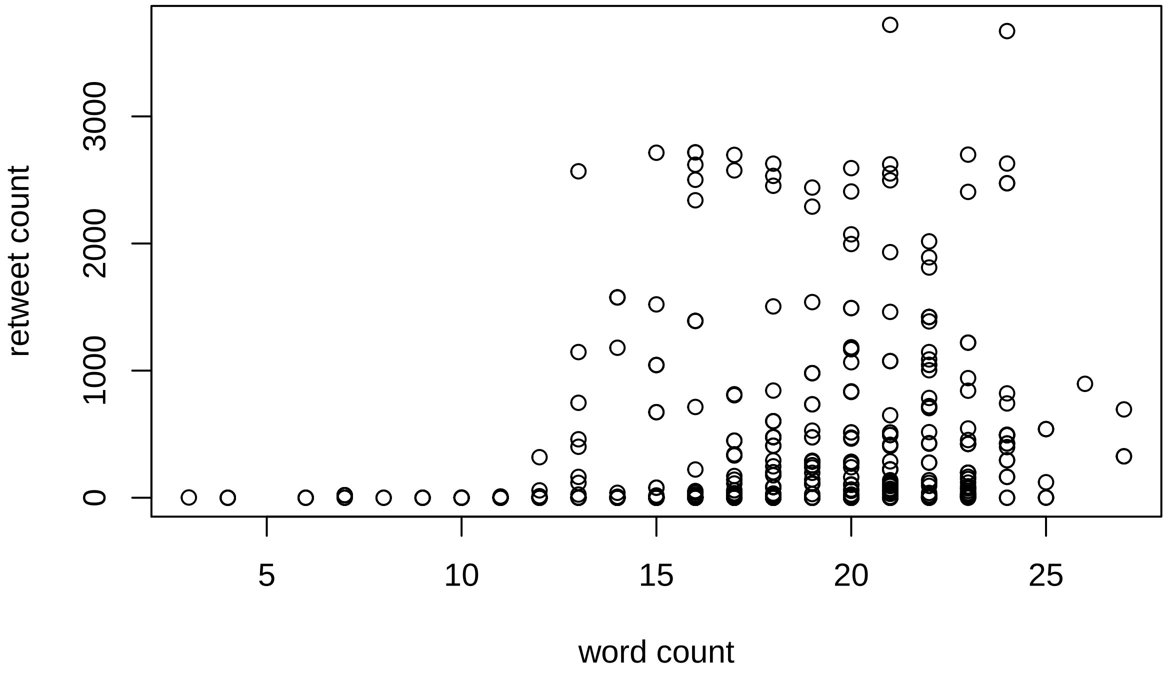
```
## Warning: Unknown or uninitialised column: 'wordCount'.
```

```
hist(tweets$wordCount)
```



Next we plot number of words and number of retweets

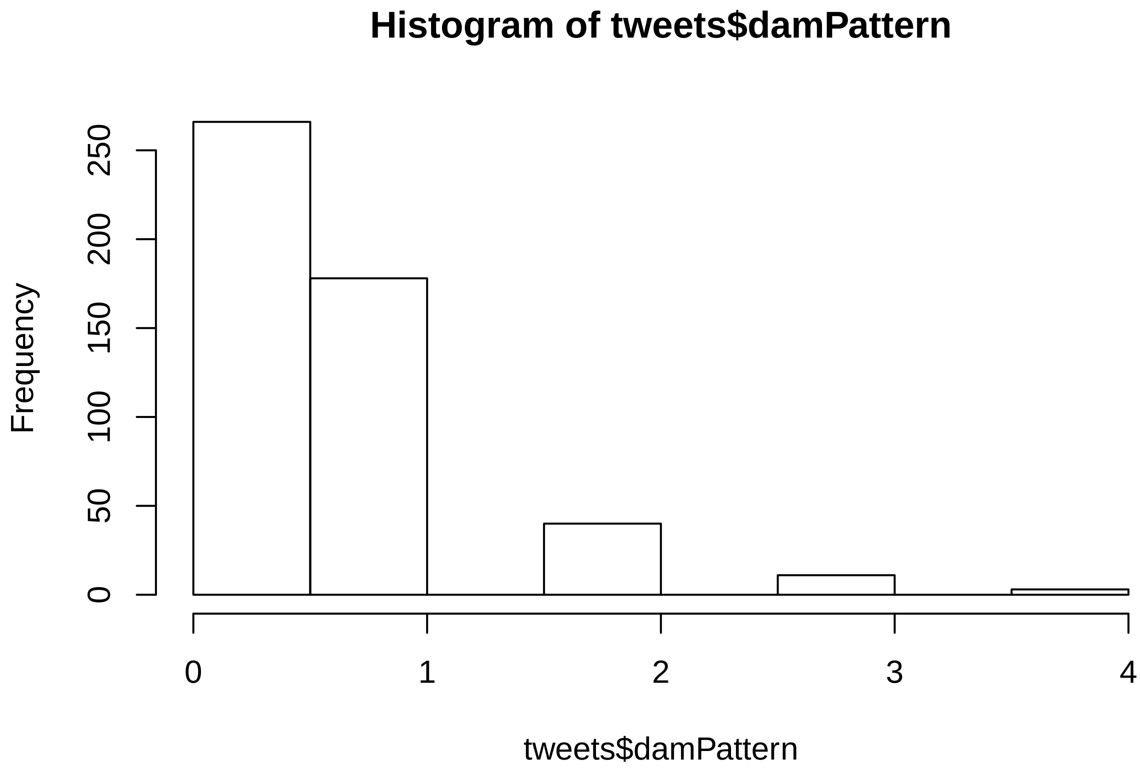
```
plot(tweets$wordCount, tweets$retweetCount, xlab = "word count", ylab = "retweet count")
```



Next we calculate the frequency of the pattern “dam” per tweet. As we can see there are only two instances where the pattern “dam” is used. As this would not make for an exciting graph, we used the string “ing” instead for plotting it against time

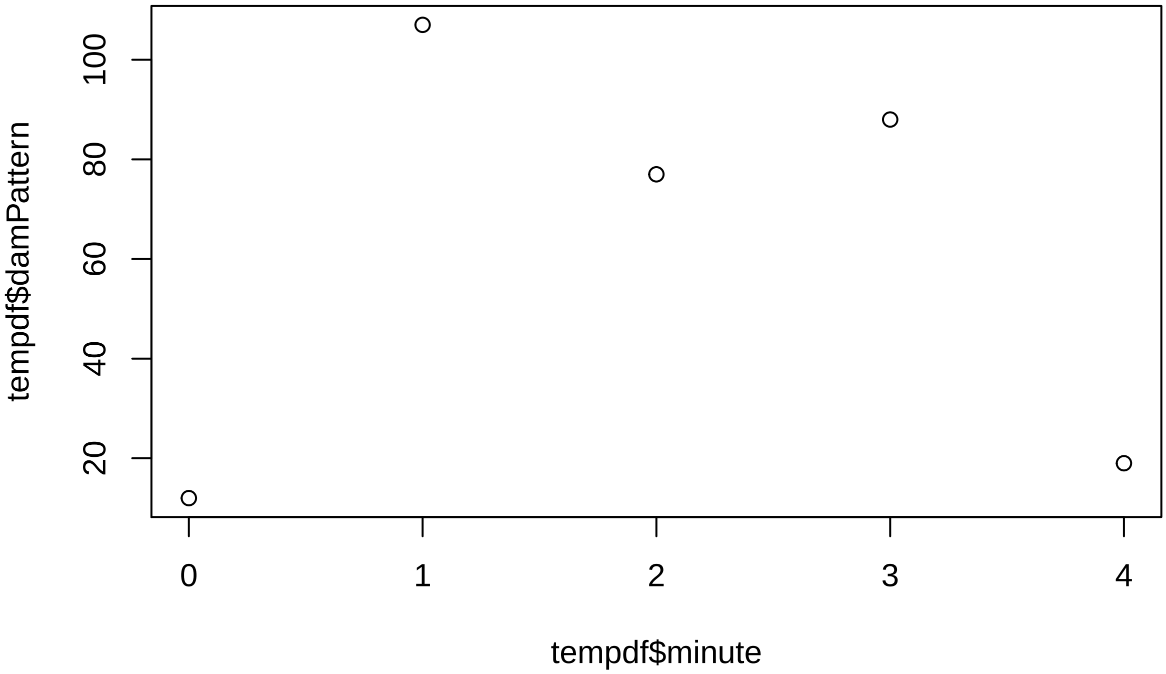
```
library(stringr)

tweets$damPattern <- str_count(tweets$text, "ing")
hist(tweets$damPattern)
```



```
tweets$realTime <- as.POSIXct(tweets$created, format = "%y-%m-%d %H:%M:%S")
tweets$minute <- as.POSIXlt(tweets$realTime)$min

tempdf <- tweets[,c("damPattern", "minute")]
tempdf <- aggregate(cbind(damPattern)~minute, data = tempdf, FUN = sum)
plot(tempdf$minute,tempdf$damPattern)
```



## Question 3

First we import the data

```
library(readr)
trainTitanic <- read_csv("C:/Users/Loic RW/Google Drive/Big Data and Business Analytics/Workshops/Sessi
                         col_types = cols(Name = col_skip(), PassengerId = col_skip()))

#fix variables
trainTitanic$Survived <- as.ordered(trainTitanic$Survived)
trainTitanic$Pclass <- as.factor(trainTitanic$Pclass)
trainTitanic$Sex <- as.factor(trainTitanic$Sex)
trainTitanic$Embarked <- as.factor(trainTitanic$Embarked)

trainTitanic$Age[is.na(trainTitanic$Age)] <- mean(trainTitanic$Age, na.rm = TRUE)
trainTitanic$Cabin <- NULL
trainTitanic$Ticket <- NULL

trainTitanic <- trainTitanic[complete.cases(trainTitanic),]

trainTitanic$Age <- scale(trainTitanic$Age)
trainTitanic$Fare <- scale(trainTitanic$Fare)
```

Then we perform the analysis

```r
library(C50)
library(e1071)

mdl <- Survived ~ .

#Xval
#cross validation
nFolds = 10
myFolds <- cut(seq(1,nrow(trainTitanic)),
               breaks = nFolds,
               labels = FALSE)

#initialise accuracy variables
accNB <- rep(NA, nFolds)
accSVM <- rep(NA, nFolds)

specNB <- rep(NA, nFolds)
specSVM <- rep(NA, nFolds)

sensNB <- rep(NA, nFolds)
sensSVM <- rep(NA, nFolds)

AUCNB <- rep(NA, nFolds)
AUCSVM <- rep(NA, nFolds)

library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
for (i in 1:nFolds) {
  cat("Analysis of fold", i, "\n")

  #define training and test set (print commands are to help troubleshoot
  #and determine where the programme aborted during errors)
  testIndex <- which(myFolds == i, arr.ind = TRUE)
  crossTest <- trainTitanic[testIndex, ]
  crossTrain <- trainTitanic[-testIndex, ]
  print("data allocated")
  #train the models
  rsltNB <- naiveBayes(mdl, data = crossTrain)
  print("Naive Bayes trained")
  rsltSVM <- svm(mdl, data = crossTrain)
  print("svm trained")


  #predict values
  pdNB <- as.ordered(predict(rsltNB, crossTest, type = "class"))
  print("nb predicted")
  pdSVM = as.ordered(predict(rsltSVM, crossTest))
  print("svm predicted")
  pdNB
  pdSVM

  #measure accuracy
  accNB[i] = mean(pdNB == crossTest$Survived)
  print("nb accuracy saved")
  accSVM[i] = mean(pdSVM == crossTest$Survived)
  print("SVM accuracy saved")

  confMatrix <- table(pdNB, crossTest$Survived)
  specNB[i] = specificity(confMatrix)
  sensNB[i] = sensitivity(confMatrix)
  print("nb other measures done")

  confMatrix <- table(pdSVM, crossTest$Survived)
  specSVM[i] = specificity(confMatrix)
  sensSVM[i] = sensitivity(confMatrix)
  print("svm other measures done")

  #AUC
  crossTrain$Survived <- as.numeric(crossTrain$Survived)
  crossTest$Survived <- as.numeric(crossTest$Survived)

  rsltNB <- naiveBayes(mdl, data = crossTrain)
  print("Naive Bayes trained")
  rsltSVM <- svm(mdl, data = crossTrain)
  print("svm trained")

  pdNB <- predict(rsltNB, crossTest, type = "raw")
  pdNB <- pdNB[,2]
  print("nb predicted")
  pdSVM = predict(rsltSVM, crossTest)
  print("svm predicted")


  AUCNB[i] = as.numeric(auc(crossTest$Survived, pdNB))
  AUCSVM[i] = as.numeric(auc(crossTest$Survived, pdSVM))
}
```

```
## Analysis of fold 1
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 2
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 3
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 4
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 5
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 6
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 7
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 8
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 9
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## Analysis of fold 10
## [1] "data allocated"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
## [1] "nb accuracy saved"
## [1] "SVM accuracy saved"
## [1] "nb other measures done"
## [1] "svm other measures done"
## [1] "Naive Bayes trained"
## [1] "svm trained"
## [1] "nb predicted"
## [1] "svm predicted"
```

```
#determine the average accuracy over the 10 folds for each model
avgAccNB = mean(accNB)
avgAccSVM = mean(accSVM)

avgSpecNB = mean(specNB)
avgSpecSVM = mean(specSVM)

avgSensNB = mean(sensNB)
avgSensSVM = mean(sensSVM)

avgAUCNB = mean(AUCNB)
avgAUCSVM = mean(AUCSVM)

#print all the results
avgAccNB
```

```
## [1] 0.7761747
```

```
avgAccSVM
```

```
## [1] 0.8245148
```

```r
avgSpecNB
```

```
## [1] 0.5849951
```

```r
avgSpecSVM
```

```
## [1] 0.7150887
```

```r
avgSensNB
```

```
## [1] 0.8963486
```

```r
avgSensSVM
```

```
## [1] 0.8898537
```

```r
avgAUCNB
```

```
## [1] 0.823757
```

```r
avgAUCSVM
```

```
## [1] 0.8396036
```

Now we will examine the ROC

```r
#Xval
library(caret)
library(ROSE)
```

```
## Loaded ROSE 0.0-3
```

```r
library(caTools)

set.seed(12345)

trainTitanic$Survived <- as.numeric(trainTitanic$Survived)

sample <- sample.split(trainTitanic$Survived, SplitRatio = 0.8)

train = subset(trainTitanic, sample == TRUE)
test = subset(trainTitanic, sample == FALSE)
nrow(test)
```

```
## [1] 178
```

```r
#train the models
rsltNB <- naiveBayes(mdl, data = train)
print("Naive Bayes trained")
```

```
## [1] "Naive Bayes trained"
```

```r
rsltSVM <- svm(mdl, data = train)
print("svm trained")
```

```
## [1] "svm trained"
```

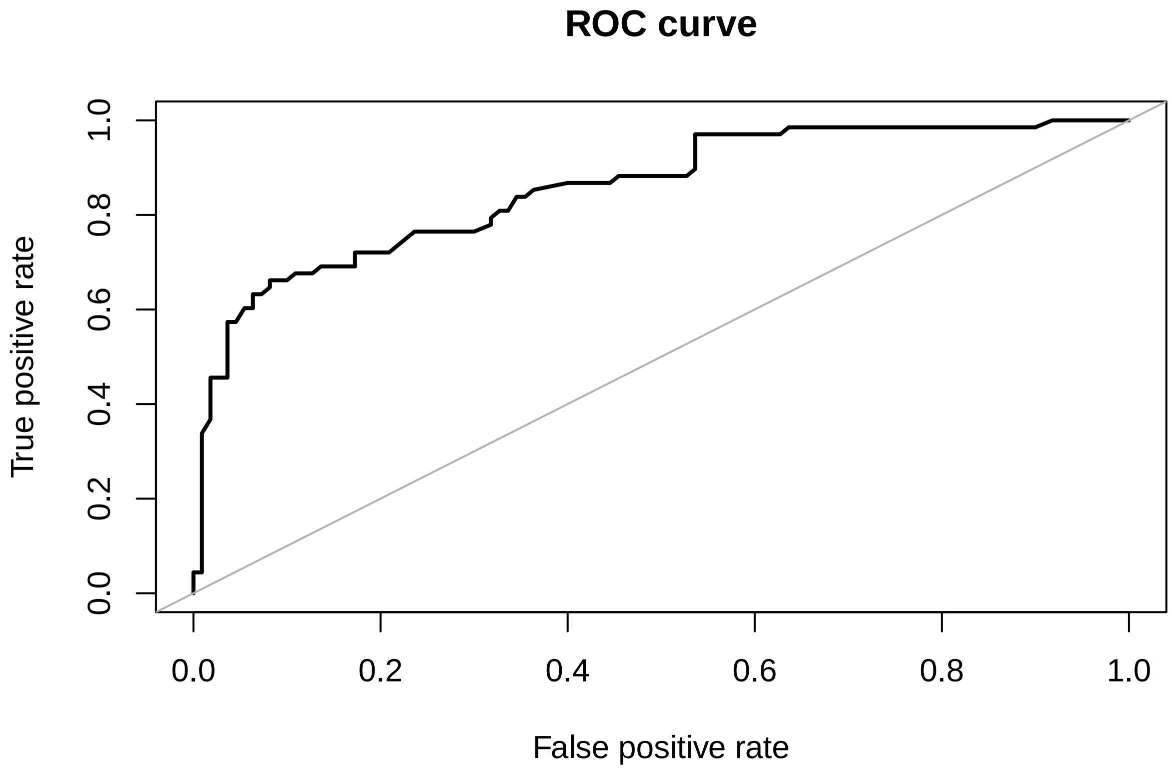
```r
#predict values
pdNB <- predict(rsltNB, test, type = "raw")
pdNB <- pdNB[,2]
print("nb predicted")
```

```
## [1] "nb predicted"
```

```r
pdSVM = predict(rsltSVM, test)
print("svm predicted")
```

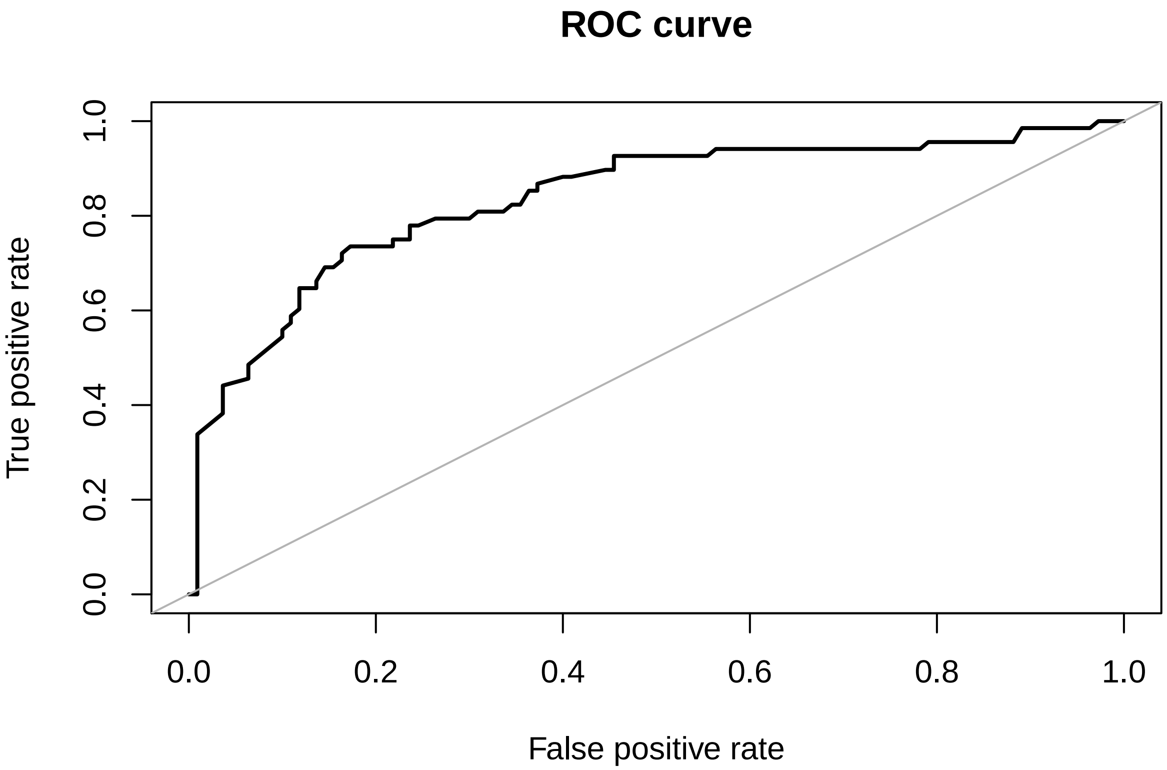
```
## [1] "svm predicted"
```

```r
roc.curve(test$Survived, pdNB)
```



```
## Area under the curve (AUC): 0.856
```

```
roc.curve(test$Survived, pdSVM)
```



```
## Area under the curve (AUC): 0.837
```