SMS Spam Collection

Shiva Sankar Modala

2023-03-28

```
library(readxl)

## Warning: package 'readxl' was built under R version 4.2.3

library(tm)

## Warning: package 'tm' was built under R version 4.2.3

## Loading required package: NLP

#install.packages("SnowballC")
library(SnowballC)
library(e1071)

## Warning: package 'e1071' was built under R version 4.2.3

library(caret)

## Warning: package 'caret' was built under R version 4.2.3

## Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:NLP':
##
##     annotate

## Loading required package: lattice

# Load the SMS Spam Collection sample dataset
SpamData = read.csv("C:/Users/shiva/OneDrive/Desktop/dpa
Assignments/Assignment
4/smsspamcollection/SMSSpamCollection",sep="\t",header=FALSE,quote="",strings
AsFactors=FALSE)
colnames(SpamData) <- c("Class", "Messages")
smsCorpus <- Corpus(VectorSource(SpamData$Messages))

# Use the tm package to create a Corpus of documents
cleaningSpamData <- function(data){
  data <- tm_map(data, tolower)   # a) Convert Lowercase
  data <- tm_map(data, removeWords,stopwords("english"))  # b) Remove
stopwords,
  data <- tm_map(data,stripWhitespace)    #  c) Strip whitespace,
```

```r
  data <- tm_map(data, removePunctuation) #  d) Remove punctuation
}
transformedData <- cleaningSpamData(smsCorpus)
```

```
## Warning in tm_map.SimpleCorpus(data, tolower): transformation drops
documents
```

```
## Warning in tm_map.SimpleCorpus(data, removeWords, stopwords("english")):
## transformation drops documents
```

```
## Warning in tm_map.SimpleCorpus(data, stripWhitespace): transformation
drops
## documents
```

```
## Warning in tm_map.SimpleCorpus(data, removePunctuation): transformation
drops
## documents
```

```r
# Building Document Term Matrix
dataDtm <- DocumentTermMatrix(transformedData)

# Use findFreqTerms tocontruct features from words occuring more than 10
times
df_new <- findFreqTerms(dataDtm, lowfreq = 10)
sparse <- removeSparseTerms(dataDtm, 0.99)
sparse
```

```
## <<DocumentTermMatrix (documents: 5574, terms: 117)>>
## Non-/sparse entries: 14050/638108
## Sparsity           : 98%
## Maximal term length: 9
## Weighting          : term frequency (tf)
```

```r
smsSparse <- as.data.frame(data.matrix((sparse)))

smsSparse$class <- SpamData$Class
smsSparse$class <- as.factor(smsSparse$class)

# proceed to split the data into a training and test set - for each create a
DocumentTermMatrix
set.seed(12345)
index <- createDataPartition(smsSparse$class, p = 0.8, list= FALSE)
trainSms <- smsSparse[index,]
testSms <- smsSparse[-index,]

#  convert the DocumentTermMatrix train/test matrices to a Boolean
representation
#  fit a SVM using the e1071 package
modelSvm <- svm(class~., data = trainSms, scale = FALSE, kernel ="linear",
type = "C")
predictTrain <- predict(modelSvm, trainSms)
predictLinear <- predict(modelSvm, testSms)
```

```
accuracyTrain <- confusionMatrix(as.factor(predictTrain), as.factor(trainSms$class))
accuracyTest <-
confusionMatrix(as.factor(predictLinear),as.factor(testSms$class))

# Report your training and test set accuracy.
cat("\n Accuracy Train: ")
```

```
## 
##  Accuracy Train:
```

```
accuracyTrain
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction  ham spam
##       ham  3835  126
##       spam   27  472
## 
##                Accuracy : 0.9657
##                  95% CI : (0.9599, 0.9708)
##     No Information Rate : 0.8659
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.8412
## 
##  Mcnemar's Test P-Value : 2.322e-15
## 
##             Sensitivity : 0.9930
##             Specificity : 0.7893
##          Pos Pred Value : 0.9682
##          Neg Pred Value : 0.9459
##              Prevalence : 0.8659
##          Detection Rate : 0.8599
##    Detection Prevalence : 0.8881
##       Balanced Accuracy : 0.8912
## 
##        'Positive' Class : ham
## 
```

```
cat("\n Accuracy Test: ")
```

```
## 
##  Accuracy Test:
```

```
accuracyTest
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction ham spam
##       ham  954   39
##       spam  11  110
##                                           
##                Accuracy : 0.9551          
##                  95% CI : (0.9413, 0.9665)
##     No Information Rate : 0.8662          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.7896          
##                                           
##  Mcnemar's Test P-Value : 0.0001343       
##                                           
##             Sensitivity : 0.9886          
##             Specificity : 0.7383          
##          Pos Pred Value : 0.9607          
##          Neg Pred Value : 0.9091          
##              Prevalence : 0.8662          
##          Detection Rate : 0.8564          
##    Detection Prevalence : 0.8914          
##       Balanced Accuracy : 0.8634          
##                                           
##        'Positive' Class : ham             
## 
```