Spam or Ham message Classification

Anish Singh Walia

Requiring the necessary packages-

require(quanteda)#natural language processing package  
?quanteda   
require(RColorBrewer)  
require(ggplot2)  
require(pROC)#to plot a ROC curve and find auc

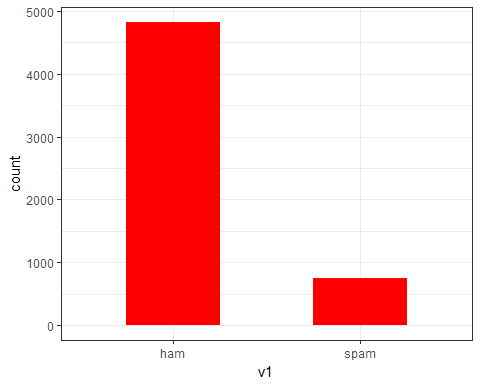
**quanteda** makes it easy to manage texts in the form of a **corpus**, defined as a collection of texts that includes document-level variables specific to each text, as well as meta-data for documents and for the collection as a whole. quanteda includes tools to make it easy and fast to manuipulate the texts in a corpus, by performing the most common natural language processing tasks simply and quickly, such as tokenizing, stemming, or forming ngrams. quanteda’s functions for tokenizing texts and forming multiple tokenized documents into a document-feature matrix are both extremely fast and extremely simple to use. quanteda can segment texts easily by words, paragraphs, sentences, or even user-supplied delimiters and tags.

#### loading the dataset

spam<-read.csv("F:/PROJECTS/Datasets/spam.csv",header=TRUE, sep=",", quote='\"\"', stringsAsFactors=FALSE)  
  
table(spam$v1)

##   
## ham spam   
## 4825 747

#checking the distribution of type of messages  
theme\_set(theme\_bw())  
ggplot(aes(x=v1),data=spam) +  
 geom\_bar(fill="red",width=0.5)



Now let’s add appropiate names to the columns.

names(spam)<-c("type","message")  
head(spam)

## type  
## 1 ham  
## 2 ham  
## 3 spam  
## 4 ham  
## 5 ham  
## 6 spam  
## message  
## 1 Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...  
## 2 Ok lar... Joking wif u oni...  
## 3 Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's  
## 4 U dun say so early hor... U c already then say...  
## 5 Nah I don't think he goes to usf, he lives around here though  
## 6 FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, å£1.50 to rcv  
## NA NA NA  
## 1   
## 2   
## 3   
## 4   
## 5   
## 6

Now we can sample the data.We can randomize our data using the sample() command.If the data is not stored in a random distribution, this will help to ensure that we are dealing with a random draw from our data. The set.seed() is to ensure reproducable results.

set.seed(2012)  
spam<-spam[sample(nrow(spam)),]

### Now let’s build the spam and ham Wordclouds

We’ll use quanteda’s corpus() command to construct a corpus from the Text field of our raw data.A corpus can be thought of as a master copy of our dataset from which we can pull subsets or observations as needed.

After this I will attach the Label field as a document variable to the corpus using the docvars() command. We attach Label as a variable directly to our corpus so that we can associate SMS messages with their respective ham/spam label later in the analysis.

?corpus #to search more on this method  
msg.corpus<-corpus(spam$message)  
docvars(msg.corpus)<-spam$type #ataching the label to the corpus message text

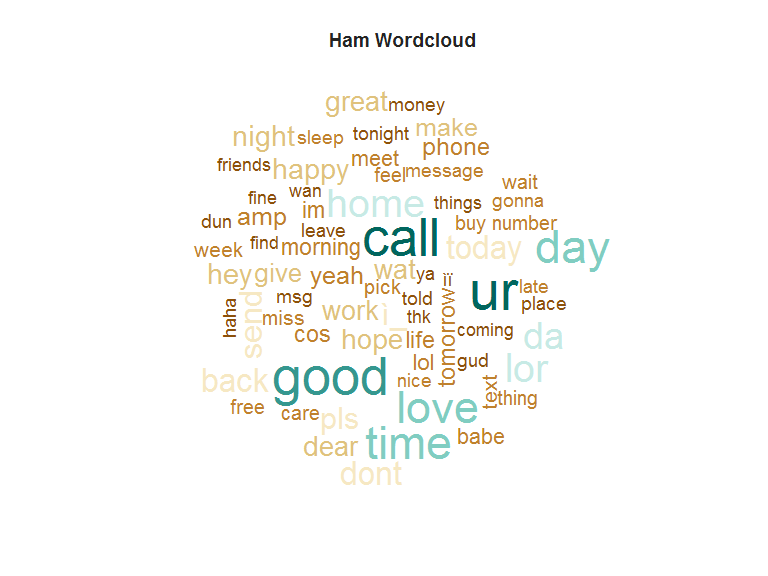
Let’s plot the wordcloud now-

#subsetting only the spam messages  
spam.plot<-corpus\_subset(msg.corpus,docvar1=="spam")  
  
#now creating a document-feature matrix using dfm()  
spam.plot<-dfm(spam.plot, tolower = TRUE, remove\_punct = TRUE, remove\_twitter = TRUE, remove\_numbers = TRUE, remove=stopwords("SMART"))  
  
spam.col <- brewer.pal(10, "BrBG")   
  
textplot\_wordcloud(spam.plot, min.freq = 16, color = spam.col)   
title("Spam Wordcloud", col.main = "grey14")



### Generating the Ham wordcloud

ham.plot<-corpus\_subset(msg.corpus,docvar1=="ham")  
ham.plot<-dfm(ham.plot,tolower = TRUE, remove\_punct = TRUE, remove\_twitter = TRUE, remove\_numbers = TRUE,remove=c("gt", "lt", stopwords("SMART")))  
ham.col=brewer.pal(10, "BrBG")   
textplot\_wordcloud(ham.plot,min.freq=50,colors=ham.col,fixed.asp=TRUE)  
title("Ham Wordcloud",col.main = "grey14")



## Prediction Using Naive Bayes Classifier

Naive Bayes classifiers are a class of simple linear classifiers which are *conditional probability* models based on **Bayes** Theoram i.e

.

The special thing about Naive Bayes classifiers are that they follow *Conditional Independence Theoram* i.e the features are uncorrelated and independent of each other which is often always crude.Secondly,they assume that the data samples are drawn from a identical and independent distribution- **IID** is the term which is famous in Statistics.

#separating Train and test data  
spam.train<-spam[1:4458,]  
spam.test<-spam[4458:nrow(spam),]  
  
msg.dfm <- dfm(msg.corpus, tolower = TRUE) #generating document freq matrix  
msg.dfm <- dfm\_trim(msg.dfm, min\_count = 5, min\_docfreq = 3)   
msg.dfm <- dfm\_weight(msg.dfm, type = "tfidf")   
  
#trining and testing data of dfm   
msg.dfm.train<-msg.dfm[1:4458,]  
  
msg.dfm.test<-msg.dfm[4458:nrow(spam),]

Training the Naive Bayes classifier-

nb.classifier<-textmodel\_NB(msg.dfm.train,spam.train[,1])  
nb.classifier

## Fitted Naive Bayes model:  
## Call:  
## textmodel\_NB.dfm(x = msg.dfm.train, y = spam.train[, 1])  
##   
##   
## Training classes and priors:  
## spam ham   
## 0.5 0.5   
##   
## Likelihoods: Class Posteriors:  
## 30 x 4 Matrix of class "dgeMatrix"  
## spam ham spam ham  
## you 5.001507e-03 0.0096798156 0.34067144 0.6593286  
## have 4.322289e-03 0.0042303673 0.50537386 0.4946261  
## 1 2.695748e-03 0.0009529526 0.73882413 0.2611759  
## new 3.492485e-03 0.0010753934 0.76457487 0.2354251  
## . 6.965338e-03 0.0168302131 0.29271598 0.7072840  
## please 2.339097e-03 0.0011593603 0.66860811 0.3313919  
## call 1.058603e-02 0.0021859571 0.82884759 0.1711524  
## i 8.439760e-04 0.0112106647 0.07001254 0.9299875  
## wait 1.860817e-04 0.0011538316 0.13887596 0.8611240  
## for 5.699340e-03 0.0045025239 0.55865674 0.4413433  
## hope 2.334040e-04 0.0017258550 0.11912872 0.8808713  
## tonight 1.137075e-04 0.0011106417 0.09287182 0.9071282  
## too 3.802754e-05 0.0017024748 0.02184860 0.9781514  
## bad 1.232420e-04 0.0006045270 0.16934219 0.8306578  
## as 1.339518e-03 0.0020699791 0.39287852 0.6071215  
## well 2.938089e-04 0.0017334850 0.14492664 0.8550734  
## but 2.528948e-04 0.0043933716 0.05442968 0.9455703  
## rock 3.802754e-05 0.0002684845 0.12406542 0.8759346  
## night 3.003905e-04 0.0017976398 0.14317739 0.8568226  
## anyway 3.802754e-05 0.0005405216 0.06572915 0.9342709  
## going 1.538819e-04 0.0023951976 0.06036762 0.9396324  
## a 7.856726e-03 0.0064918622 0.54756091 0.4524391  
## now 6.254232e-03 0.0028758075 0.68501697 0.3149830  
## good 6.723203e-04 0.0030342352 0.18138681 0.8186132  
## speak 8.003838e-04 0.0004728416 0.62862694 0.3713731  
## to 1.113210e-02 0.0075761991 0.59503541 0.4049646  
## soon 2.642059e-04 0.0010608467 0.19939274 0.8006073  
## today 1.120666e-03 0.0019041688 0.37048833 0.6295117  
## is 4.451802e-03 0.0058153446 0.43359683 0.5664032  
## accept 3.802754e-05 0.0003188419 0.10655871 0.8934413

The model outputs the Probabilities of the message being Spam or ham.

### Let’s Test the Model

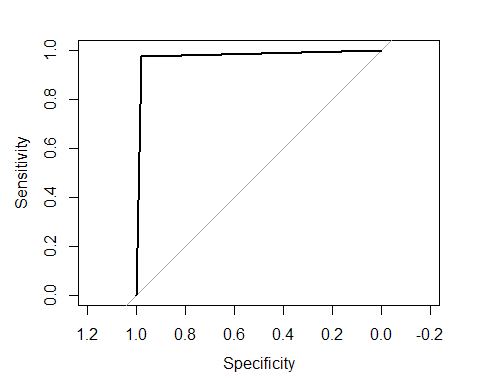
pred<-predict(nb.classifier,msg.dfm.test)  
  
#generating a confusion matrix  
  
# use pred$nb.predicted to extract the class labels  
table(predicted=pred$nb.predicted,actual=spam.test[,1])

## actual  
## predicted ham spam  
## ham 943 4  
## spam 19 149

#16 wrongly classified for ham and 7 examples wrongly classified for spam  
  
#acccuracy of the classifier on Test data  
mean(pred$nb.predicted==spam.test[,1])\*100

## [1] 97.93722

#accuracy of 97% on test set  
  
prednum<-ifelse(pred$nb.predicted=="spam",1,2)  
  
auc<-roc(as.factor(spam.test[,1]),prednum)  
plot(auc)



auc$auc

## Area under the curve: 0.9771

#Area under the curve: 0.9771

In the ROC curve the area under the curve is **0.9771** which is a very nice score and implies that the model can easily recognize text messages as either spam or ham. ROC curve is plotted between **Sensitivity**-i.e true positive rate(positive classes being classified correctly) vs the **Specificity**-i.e true negetive rate(negetive classes being clssified correctly)

In the Confusion matrix , the **diagonals** are the correctly classified examples while the **off-diagonals** the incorrectly classifiec examples.

## Conclusion

This was a simple article on classifying text messages as ham or spam using some basic natural language processing and then building a naive Bayes text classifier.I urge the readers to implement and use the knowledge acquired from this article in making their own text classifiers and solving different problems related to text processing and NLP etc. Ofcourse,there are various other packages to do text processing and building such models.

Hope you guys liked the article , make sure to like and share it.