DATA 607 - Project 4

Adam Douglas

11/2/2018

Goal

Our goal is to build a classification model that can look at example emails and determine, from the text contained within, whether the email is unsolicited email (a.k.a. spam) or if it not ("ham").

Example Files

We begin with two directories. Each directory contains examples of either unsolicited (i.e. "spam") emails or actual emails (aka "ham"). Our first step is to load each of these documents into vectors and then into an object known as a corpus, which contains the documents' text as well as metadata about each.

```r
# Load our spam documents by getting a list of files and
# then loading each document's text into a vector.

# List the files
spam <- list.files("spam")

# Begin with the first file
docs <- unlist(str_split(read_file(paste("spam/",spam[1],sep="")),"\\n\\n",n=2))[2]

# Iterate through the rest of the list
for (i in 2:length(spam)){
  # split the body from the header
  email <- str_split(read_file(paste("spam/",spam[i],sep="")),"\\n\\n",n=2)
  # take the body only
  docs[i] <-iconv(unlist(email)[2],"ASCII","UTF-8",sub="")
}

# Remove HTML tags
docs <- str_replace_all(docs,"<.+","")

# Create a corpus from the vector
spamCorp <- VCorpus(VectorSource(docs))

# Add meta tag identifying it as spam
meta(spamCorp, "description", type="index") <- rep("spam",length(docs))
```

```r
# Repeat the same for ham documents
ham <- list.files("ham")

docs <- read_file(paste("ham/", ham[1], sep=""))

for (i in 2:length(ham)){
  email <- str_split(read_file(paste("ham/",ham[i],sep="")),"\\n\\n",n=2)
  docs[i] <- iconv(unlist(email)[2],"ASCII","UTF-8",sub="")
}
```

```r
docs <- str_replace_all(docs,"<.+","")

hamCorp <- VCorpus(VectorSource(docs))

meta(hamCorp, "description", type="index") <- rep("ham",length(docs))
```

After loading our examples, we have 2,551 examples of regular emails ("ham") and 1,398 examples of unsolicited emails ("spam").

First Look

Now that we have our corpora we can begin to look at what words are most frequently used in each type of email.

First, we should change everything to lower case, remove punctuation and any stopwords. Stopwords are common English words that will skew our frequency counts like "a", "and", or "the".

```r
# Change to lower case
spamCorp <- tm_map(spamCorp, content_transformer(str_to_lower))
hamCorp <- tm_map(hamCorp, content_transformer(str_to_lower))

# Remove punctuation
spamCorp <- tm_map(spamCorp, content_transformer(removePunctuation))
hamCorp <- tm_map(hamCorp, content_transformer(removePunctuation))

# Remove English stopwords
spamCorp <- tm_map(spamCorp, removeWords, stopwords("english"))
hamCorp <- tm_map(hamCorp, removeWords, stopwords("english"))
```

Now we can look at frequencies of words and build a word cloud to see what the most common terms are for each type of email.

```r
# Spam
spamTDM <- TermDocumentMatrix(spamCorp)
sortSpam <- sort(rowSums(as.matrix(spamTDM)),decreasing=TRUE)
wordsSpam <- data.frame(word = names(sortSpam),freq=sortSpam)

# Ham
hamTDM <- TermDocumentMatrix(hamCorp)
sortHam <- sort(rowSums(as.matrix(hamTDM)),decreasing=TRUE)
wordsHam <- data.frame(word = names(sortHam),freq=sortHam)
```

Ham

```r
# Build the ham cloud
wordcloud(words = wordsHam$word, freq = wordsHam$freq, min.freq = 150,
          max.words=200, random.order=FALSE,  rot.per=0.40,
          colors=brewer.pal(8, "Set1"))
```

Spam

```
wordcloud(words = wordsSpam$word, freq = wordsSpam$freq, min.freq = 150,
          max.words=200, random.order=FALSE, rot.per=0.40,
          colors=brewer.pal(8, "Set1"))
```

Looking at the clouds, you can definitely see the different terms used in ham and spam (respectively) and see why some emails may be easy to identify as spam at a glance.

Analysis

Now we combine our corpora into a single corpus and begin to build a classifer that will accurately tell the difference between ham and spam.

First, we combine the corpora into a single one:

```
# Combine both corpora
corpus <- c(spamCorp, hamCorp)
```

Once we have our single corpus, we can stem our words for better analysis.

```
# Stem our remaining terms
corpus <- tm_map(corpus, stemDocument)
```

Classification Model

Now we look at splitting our data set into a training and a testing dataset, creating a document term matrix, and finally creating a classification model which can categorize emails as either spam or ham.

```
# First, randomize the data in the corpus so each type of document
# are not all lumped together
corpus <- sample(corpus, length(corpus))
```

```r
head(meta(corpus,"description"),10)
```

```
##      description
## 3849         ham
## 2206         ham
## 3001         ham
## 3269         ham
## 1421         ham
## 255         spam
## 3863         ham
## 3639         ham
## 3390         ham
## 539         spam
```

```r
# Build our DTM
dtm <- DocumentTermMatrix(corpus)
```

```r
# Specifiy our train and test sets as a percentage of records
pctTrain <- 0.70
trainStop <- floor(length(corpus) * pctTrain)

labels <- unlist(meta(corpus,"description"))

# Build a container for use in RTextTools package
container <- create_container(dtm, labels = labels,
                              trainSize = 1:trainStop,
                              testSize = (trainStop+1):length(corpus),
                              virgin = FALSE)
```

Now we train the classification model. We'll use a support vector machine (SVM) and see how well it does:

```r
# Train the model
spamModel <- train_model(container, "SVM")

# Classify the model
spamResults <- classify_model(container, spamModel)

head(spamResults)
```

```
##   SVM_LABEL  SVM_PROB
## 1      spam 0.9967078
## 2       ham 0.9979081
## 3      spam 0.9999932
## 4       ham 0.7550968
## 5      spam 0.9878595
## 6       ham 0.9909539
```

Let's see how well we did classifying the test set.

```r
# Put the true label and the predicted label into a data frame
output <- data.frame(
  labels = labels[(trainStop+1):length(corpus)],
  predicted = spamResults$SVM_LABEL
)

# Compare true vs. predicted
```

Table 1: Actual vs. Predicted

Actual	Predicted	
	ham	spam
ham	758	11
spam	25	391

```
kable(table(output), caption="Actual vs. Predicted") %>%
  kable_styling(bootstrap_options = "striped",
                full_width = FALSE, position = "left") %>%
  add_header_above(c("Actual"= 1,"Predicted"=2))
```

We see nearly a 97% accuracy in the classification, which is pretty good. Even better, on the few errors we make, we tend to flag spam as ham rather than flagging ham as spam.