lab7

Credit: materials adapted from Patrick Chester, with some examples taken from Ken Benoit's NYU Dept. of Politics short course Fall 2014

```r
# Clear Global Environment
rm(list = ls())

# Setting WD
setwd("/Users/Lingyi/TAD/lab/Text-as-Data-Lab-Spr2018/W7_03_08_18/")

# Installing / Loading Libraries
# install.packages("tm")
# install.packages("NLP")
# install.packages("https://cran.r-project.org/bin/windows/contrib/3.4/prodlim_1.6.1.zip",
#repos = NULL, method = "libcurl")
#install.packages("RTextTools")

library(NLP)
library(tm)
library(RTextTools)
```

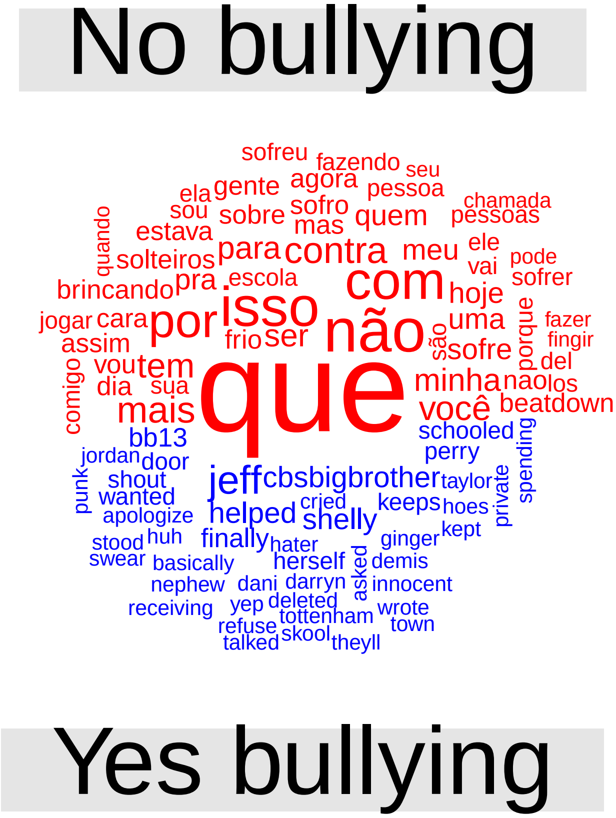
```
## Loading required package: SparseM

##
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':
##
##     backsolve
```

```r
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

1 Visualizing Bullying Data—Example from Pablo Barbera's Short Course on R, NYU 2016

```r
# https://github.com/pablobarbera/data-science-workshop

df.tweets <- read.csv("bullying.csv", stringsAsFactors = F)

# Identify posts with and without bullying traces and create large documents
no_bullying <- paste(df.tweets$text[df.tweets$bullying_traces=="n"], collapse=" ")
yes_bullying <- paste(df.tweets$text[df.tweets$bullying_traces=="y"], collapse=" ")

# Create DTM and preprocess
groups <- VCorpus(VectorSource(c("No bullying" = no_bullying, "Yes bullying" = yes_bullying)))
groups <- tm_map(groups, content_transformer(tolower))
groups <- tm_map(groups, removePunctuation)
groups <- tm_map(groups, stripWhitespace)
bullying_dtm <- DocumentTermMatrix(groups)

# Label the two groups
bullying_dtm$dimnames$Docs = c("No bullying", "Yes bullying")

# Transpose matrix so that we can use it with comparison.cloud
bullying_tdm <- t(bullying_dtm)

# Compute TF-IDF transformation
bullying_tdm <- as.matrix(weightTfIdf(bullying_tdm))

# Display the two word clouds
comparison.cloud(bullying_tdm, max.words=100, colors=c("red", "blue"))
```



```
# Function is from the wordcloud package
```

2 Classification with SVM

```
# A) Linear - whole sample

# Let's train an SVM
df.tweets$type <- as.numeric(factor(df.tweets$bullying_traces))

# New package, better for SVM
```

```r
#?create_matrix
bullying_dfm  <- create_matrix(df.tweets$text,
                      language="english",
                      stemWords = FALSE,
                      weighting = weightTfIdf,
                      removePunctuation = FALSE
                      )
str(bullying_dfm)
```

```
## List of 6
##  $ i       : int [1:45029] 1 1 1 1 1 1 1 1 1 1 ...
##  $ j       : int [1:45029] 1381 1698 1841 2470 3586 4250 4272 5097 7251 7533 ...
##  $ v       : Named num [1:45029] 1.1176 0.0969 1.1176 1.0267 0.9736 ...
##   ..- attr(*, "names")= chr [1:45029] "1" "1" "1" "1" ...
##  $ nrow    : int 5022
##  $ ncol    : int 11824
##  $ dimnames:List of 2
##   ..$ Docs : chr [1:5022] "#Canada News: And the saga of official bullying #bilingualism continues.
##   ..$ Terms: chr [1:11824] "" "" "" "" ...
##  - attr(*, "class")= chr [1:2] "DocumentTermMatrix" "simple_triplet_matrix"
##  - attr(*, "weighting")= chr [1:2] "term frequency - inverse document frequency (normalized)" "tf-idf
```

```r
# Make it all in-sample
#?create_container
container <- create_container(bullying_dfm,
                              t(df.tweets$type),
                              trainSize = 1:length(df.tweets$type),
                              virgin = FALSE
                              )

# Train the model
#?cross_validate
cv.svm <- cross_validate(container, nfold = 2, algorithm = 'SVM', kernel = 'linear')
```

```
## Fold 1 Out of Sample Accuracy = 0.9002997
## Fold 2 Out of Sample Accuracy = 0.8843024
```

```r
## Comments:
# linear vs radial kernels, radial can overfit
# linear kernel is faster
# nfold is the number of times you have a different test set

# B) Linear - 90% Training data

training_break <- as.integer(0.9*nrow(df.tweets))

container      <- create_container(bullying_dfm,
                                   t(df.tweets$type),
                                   trainSize = 1:training_break,
                                   testSize = (training_break+1):nrow(df.tweets),
                                   virgin = FALSE
                                   )

# Let's train the model
cv.svm <- cross_validate(container,
                         nfold = 4,
                         algorithm = 'SVM',
                         kernel = 'linear'
                         )
```

```
## Fold 1 Out of Sample Accuracy = 0.7858268
## Fold 2 Out of Sample Accuracy = 0.780083
## Fold 3 Out of Sample Accuracy = 0.8103586
## Fold 4 Out of Sample Accuracy = 0.8003096
```

```
# Validate
cv.svm$meanAccuracy
```

```
## [1] 0.7941445
```

```
prop.table(table(df.tweets$type)) # baseline
```

```
##
##         1         2
## 0.7369574 0.2630426
```

```
# How well did we do?

# C) Radial - 90% training data

# Let's try again with the radial kernel
cv.svm <- cross_validate(container,
                         nfold = 4,
                         algorithm = 'SVM',
                         kernel = 'radial'
                         )
```

```
## Fold 1 Out of Sample Accuracy = 0.730738
## Fold 2 Out of Sample Accuracy = 0.7667785
## Fold 3 Out of Sample Accuracy = 0.7333861
## Fold 4 Out of Sample Accuracy = 0.7419355
```

```
cv.svm$meanAccuracy
```

```
## [1] 0.7432095
```

```
# D) Linear - 50% training data

# What if we try with different % test/train?
training_break <- as.integer(0.5*nrow(df.tweets))

# There is no theoretical reason to choose .5 or .9
container      <- create_container(bullying_dfm,
                                   t(df.tweets$type),
                                   trainSize = 1:training_break,
                                   testSize = (training_break+1):nrow(df.tweets),
                                   virgin = FALSE
)
cv.svm$meanAccuracy
```

```
## [1] 0.7432095
```

```r
prop.table(table(df.tweets$type)) # baseline
```

```
## 
##         1         2 
## 0.7369574 0.2630426
```

3 Virality of stories from NYT

```r
nyt.fb <- read.csv("nyt-fb.csv", stringsAsFactors = FALSE)

#str(nyt.fb)


# Create variables for month and hour
#head(nyt.fb$created_time)

month <- substr(nyt.fb$created_time, 6, 7)

hour <- substr(nyt.fb$created_time, 12, 13)

nyt.fb <- data.frame(nyt.fb, month, hour)


# Create a "viral" index
total.resp <- nyt.fb$likes_count + nyt.fb$shares_count + nyt.fb$comments_count

# Look at the extreme of the distribution
perc_90 <- quantile(total.resp, .9)

# Create a binary y variable with values 2 being viral and 1 being non-viral
nyt.fb$viral <- as.numeric(total.resp > perc_90)

# For the purposes of not destroying my laptop, let's choose a set of features
training_break <- as.integer(0.9*nrow(nyt.fb))

# A) Classification with SVM
nyt_dtm       <- create_matrix(nyt.fb$message, language="english", stemWords = FALSE,
                           weighting = weightTfIdf, removePunctuation = FALSE)

container      <- create_container(nyt_dtm, t(nyt.fb$type), trainSize=1:training_break,
                                   testSize=(training_break+1):nrow(nyt.fb), virgin=FALSE)

cv.svm <- cross_validate(container, nfold = 2, algorithm = 'SVM', kernel = 'linear')
```

```
## Fold 1 Out of Sample Accuracy = 0.9536051
## Fold 2 Out of Sample Accuracy = 0.9540253
```

```r
cv.svm$meanAccuracy
```

```
## [1] 0.9538152
```

```r
prop.table(table(nyt.fb$viral))
```

```
## 
```

```
##   0   1
## 0.9 0.1
```

```
# B) Classification with logistic regression

message <- removePunctuation(tolower(nyt.fb$message))
nyt.fb$israel <- grepl("israel", message)
nyt.fb$trump <- grepl("trump", message)
nyt.fb$hillary <- grepl("hillary", message)
nyt.fb$obama <- grepl("barack|obama", message)
nyt.fb$terror <- grepl("terror|isis|isil|qaeda", message)
nyt.fb$kill <- grepl("kill|murder|shot", message)
nyt.fb$debate <- grepl("debat", message)

# Fitting a logistic model
glm.viral <- glm(as.factor(viral) ~ month + hour  +
                   israel + trump + hillary + obama + terror + kill +
                   debate , data=nyt.fb, family=binomial(logit))


tab <- table(round(glm.viral$fitted.values),nyt.fb$viral)

# Accuracy of Logistic Regression
sum(diag(tab))/sum(tab)
```

```
## [1] 0.8998996
```

```
# In this case, SVM had a higher level of accuracy. Why might that be?
```