Tweets

Youhee Kil

10/18/2017

```
#install.packages(c("devtools", "rjson", "bit64", "httr"))

# restart R session !

#library(devtools)
#library("devtools", lib.loc="/Users/wschram/Library/R/3.1/library")
#devtools::install_github("hadley/httr")
#library("httr", lib.loc="/Users/wschram/Library/R/3.1/library") # GitHub httr
#library(httr)
#library(twitteR)


library("base64enc")
library("twitteR")
library("ROAuth")
library("devtools")
library("memoise")
library("whisker")
library("rstudioapi")
library("git2r")
library("withr")
```

```
##
## Attaching package: 'withr'
```

```
## The following objects are masked from 'package:devtools':
##
##     with_collate, with_envvar, with_libpaths, with_locale,
##     with_makevars, with_options, with_par, with_path
```

```
library("rjson")
library("bit64")
```

```
## Loading required package: bit
```

```
## Attaching package bit
```

```
## package:bit (c) 2008-2012 Jens Oehlschlaegel (GPL-2)
```

```
## creators: bit bitwhich
```

```
## coercion: as.logical as.integer as.bit as.bitwhich which
```

```
## operator: ! & | xor != ==
```

```
## querying: print length any all min max range sum summary
```

```
## bit access: length<- [ [<- [[ [[<-
```

```
## for more help type ?bit
```

```
##
## Attaching package: 'bit'
```

```
## The following object is masked from 'package:git2r':
##
##     clone
## The following object is masked from 'package:base':
##
##     xor
## Attaching package bit64
## package:bit64 (c) 2011-2012 Jens Oehlschlaegel
## creators: integer64 seq :
## coercion: as.integer64 as.vector as.logical as.integer as.double as.character as.bin
## logical operator: ! & | xor != == < <= >= >
## arithmetic operator: + - * / %/% %% ^
## math: sign abs sqrt log log2 log10
## math: floor ceiling trunc round
## querying: is.integer64 is.vector [is.atomic} [length] format print str
## values: is.na is.nan is.finite is.infinite
## aggregation: any all min max range sum prod
## cumulation: diff cummin cummax cumsum cumprod
## access: length<- [ [<- [[ [[<-
## combine: c rep cbind rbind as.data.frame
## WARNING don't use as subscripts
## WARNING semantics differ from integer
## for more help type ?bit64
##
## Attaching package: 'bit64'
## The following object is masked from 'package:bit':
##
##     still.identical
## The following objects are masked from 'package:base':
##
##     :, %in%, is.double, match, order, rank
library ("httr")

##
## Attaching package: 'httr'
## The following objects are masked from 'package:git2r':
##
##     config, content
## The following object is masked from 'package:memoise':
##
##     timeout
```

```
library ("httpuv")

api_key <- "glDRp5aQdnlAbZumatVyFVxPr"

api_secret <- "ypthG3NHXUV3do9nDZUlUUuYgvBf5BsAiWSoIAKi0bPpeI6oRe"

access_token <- "918997877745008646-yVGKYanITgQA6ruRnV1RG8Vr3wDk4tC"

access_token_secret <- "rHvhmWHvVMkLeb279w8Fhr3gUqXAmZKJ2JHPnl8zdlHA7"


setup_twitter_oauth(api_key, api_secret, access_token, access_token_secret)
```

```
## [1] "Using direct authentication"
```

```
tweets <- searchTwitter('#comfortwomen', n=900)

n.tweet <- length(tweets)
# convert tweets to a data frame
tweets.df <- twListToDF(tweets)

# Text Cleaning
library(NLP)
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:httr':
##
##     content
```

```
## The following object is masked from 'package:git2r':
##
##     content
```

```
library(tm)
# build a corpus, and specify the source to be character vectors
myCorpus <- Corpus(VectorSource(tweets.df$text))
# convert to lower case
#myCorpus <- tm_map(myCorpus, content_transformer(tolower))
# remove URLs
removeURL <- function(x) gsub("http[^[:space:]]*", "", x)
myCorpus <- tm_map(myCorpus, content_transformer(removeURL))
# remove anything other than English letters or space
removeNumPunct <- function(x) gsub("[^[:alpha:][:space:]]*", "", x)
myCorpus <- tm_map(myCorpus, content_transformer(removeNumPunct))
# remove stopwords
myStopwords <- c(setdiff(stopwords('english'), c("r", "big")),
                 "use", "see", "used", "via", "amp")
myCorpus <- tm_map(myCorpus, removeWords, myStopwords)
# remove extra whitespace
myCorpus <- tm_map(myCorpus, stripWhitespace)
# keep a copy for stem completion later
myCorpusCopy <- myCorpus

#STEMING
```

```
myCorpus <- tm_map(myCorpus, stemDocument) # stem words
writeLines(strwrap(myCorpus[[190]]$content, 60))
```

```
## Comfort Women emot just logic Queen NYC Glendal
## Comfortwomen PalisadesPark BergenCounti Losangel
```

```
## r refer card data mine now provid link packag cran packag
## mapreduc hadoop ad
stemCompletion2 <- function(x, dictionary) {
  x <- unlist(strsplit(as.character(x), " "))
  x <- x[x != ""]
  x <- stemCompletion(x, dictionary=dictionary)
  x <- paste(x, sep="", collapse=" ")
  PlainTextDocument(stripWhitespace(x))
}
myCorpus <- lapply(myCorpus, stemCompletion2, dictionary=myCorpusCopy)
myCorpus <- Corpus(VectorSource(myCorpus))
writeLines(strwrap(myCorpus[[190]]$content, 60))
```

```
## list(content = "Comfort Women emotion just logic Queens NYC
## Glendale Comfortwomen PalisadesPark Losangeles", meta =
## list(author = character(0), datetimestamp = list(sec =
## 33.2734460830688, min = 29, hour = 10, mday = 9, mon = 0,
## year = 118, wday = 2, yday = 8, isdst = 0), description =
## character(0), heading = character(0), id = character(0),
## language = character(0), origin = character(0)))
```

```
## r reference card data miner now provided link package cran
## package mapreduce hadoop add
```

```
# count word frequence
wordFreq <- function(corpus, word) {
  results <- lapply(corpus,
                    function(x) { grep(as.character(x), pattern=paste0("\\<",word)) }
  )
  sum(unlist(results))
}
n.miner <- wordFreq(myCorpusCopy, "japan")
n.mining <- wordFreq(myCorpusCopy, "korea")
cat(n.miner, n.mining)
```

```
## 2 5
```

```
## 9 104
# replace oldword with newword
replaceWord <- function(corpus, oldword, newword) {
  tm_map(corpus, content_transformer(gsub),
         pattern=oldword, replacement=newword)
}
myCorpus <- replaceWord(myCorpus, "miner", "mining")
myCorpus <- replaceWord(myCorpus, "universidad", "university")
myCorpus <- replaceWord(myCorpus, "scienc", "science")

myCorpus <- tm_map(myCorpus, removeNumbers)
```

```
12 / 40
```

```
## [1] 0.3
```

```
#Build Term Document Matrix

tdm <- TermDocumentMatrix(myCorpus,control = list(wordLengths = c(1, Inf)))

(freq.terms <- findFreqTerms(tdm, lowfreq = 20))
```

```
##   [1] "author"        "character"     "comfortwomen"  "content"
##   [5] "datetimestamp" "description"   "heading"       "hour"
##   [9] "id"            "isdst"         "language"      "list"
##  [13] "mday"          "meta"          "min"           "mon"
##  [17] "origin"        "rt"            "sec"           "wday"
##  [21] "yday"          "year"          "comfort"       "japan"
##  [25] "justice"       "southkorea"    "women"         "agreement"
##  [29] "korean"        "us"            "california"    "glendale"
##  [33] "memorial"      "palisadespark" "prostitute"    "ny"
##  [37] "nyc"           "queens"        "tokyo"         "usa"
##  [41] "deal"          "korea"         "sex"           "slave"
##  [45] "south"         "continue"      "lie"           "renegotiate"
##  [49] "said"          "the"           "repost"        "vietnamese"
##  [53] "now"           "high"          "never"         "newton"
##  [57] "people"        "school"        "back"          "recruited"
##  [61] "former"        "klaitb"        "president"     "soldier"
##  [65] "issue"         "brokers"       "japanese"      "in"
##  [69] "kill"          "war"           "not"           "sold"
##  [73] "money"         "whistleb"      "fairfax"       "va"
##  [77] "veterans"      "vietnam"       "virginia"      "a"
##  [81] "woman"         "father"        "kim"           "times"
##  [85] "truth"         "report"        "fortlee"       "i"
##  [89] "work"          "abducted"      "deceived"      "huh"
##  [93] "jpn"           "busted"        "fraud"         "police"
##  [97] "gov"           "coerced"       "private"       "buy"
## [101] "enough"        "houses"        "made"          "moon"
## [105] "okju"          "compensation"  "s"             "nj"
## [109] "yasukuni"      "milpitas"      "called"
```

```
term.freq <- rowSums(as.matrix(tdm))

#sentiment analysis with with tweets words about comfort women in twitter.
dd <- data.frame(term.freq)

## extract the words from the column, make it as data, to have common variable .

term.freq <- subset(term.freq, term.freq >= 20)
df <- data.frame(term = names(term.freq), freq = term.freq)
nrow(df)
```

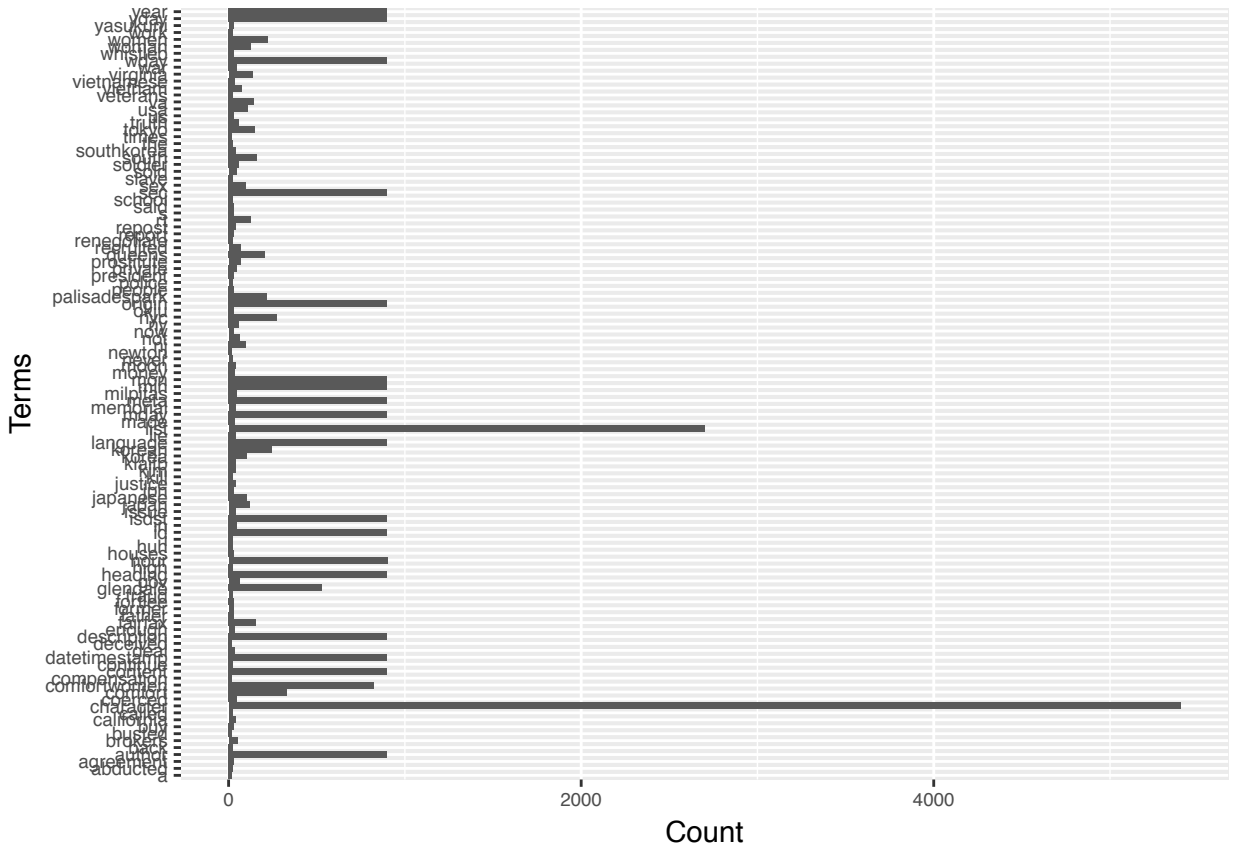
```
## [1] 111
```

```
library(ggplot2)
```

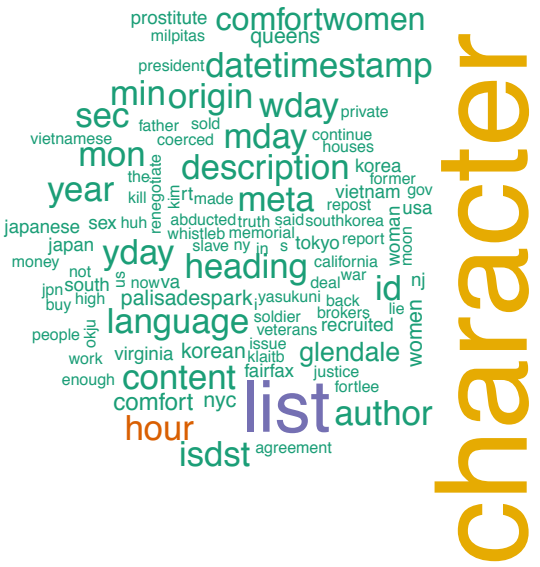
```
##
## Attaching package: 'ggplot2'
```

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

```
ggplot(df, aes(x=term, y=freq)) + geom_bar(stat="identity") +
  xlab("Terms") + ylab("Count") + coord_flip() +
  theme(axis.text=element_text(size=7))
```



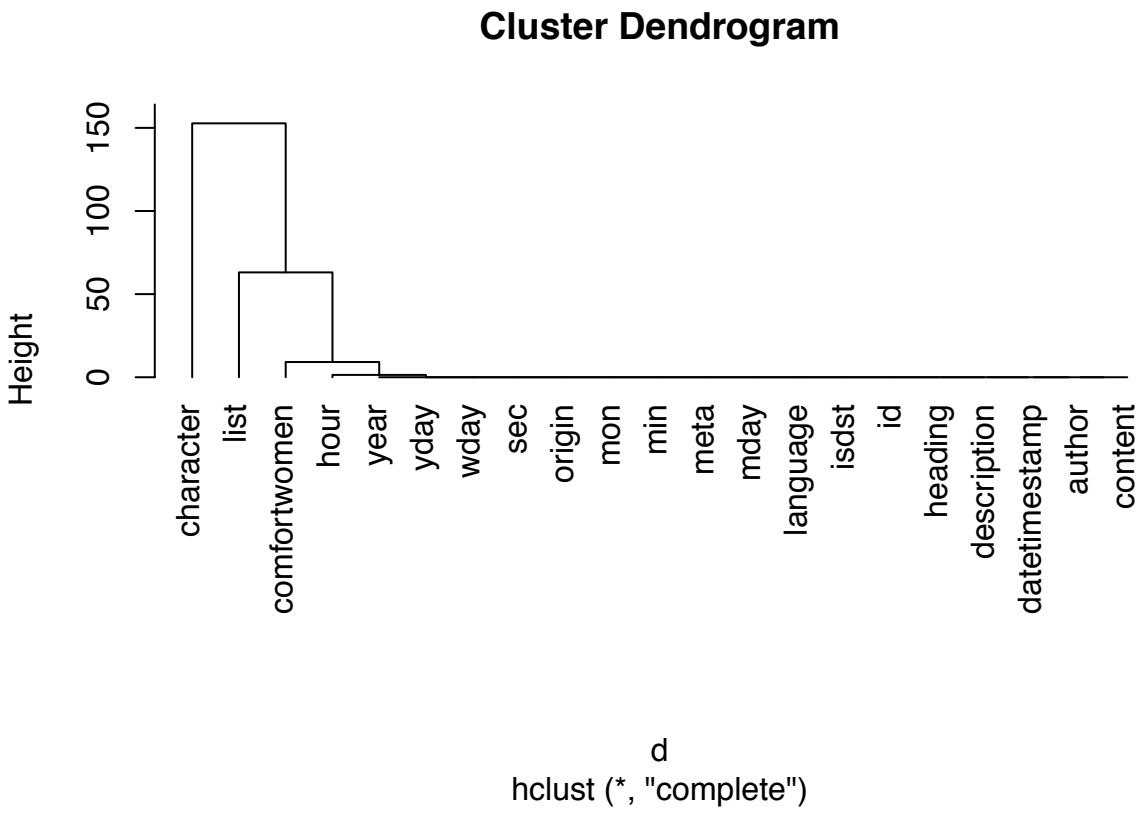

```
m <- as.matrix(tdm)
# calculate the frequency of words and sort it by frequency
word.freq <- sort(rowSums(m), decreasing = T)
# colors
#pal <- brewer.pal(9, "BuGn")[-(1:4)]
# plot word cloud
library(RColorBrewer)
library(wordcloud)

set.seed(142)
dark2 <- brewer.pal(6, "Dark2")
wordcloud(names(word.freq), word.freq, max.words=100, colors=dark2)
```



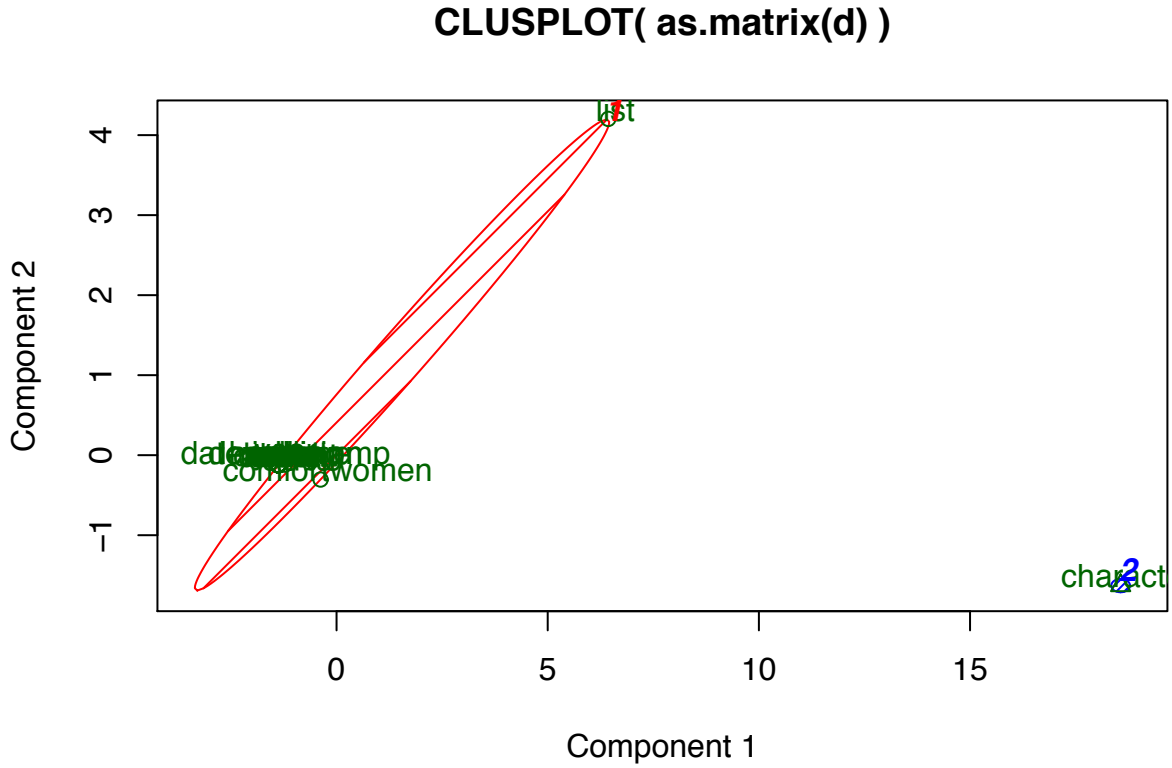

```
#Modeling
dtm <- as.DocumentTermMatrix(tdm)
library(topicmodels)
lda <- LDA(dtm, k = 8) #find 8 topics
term <- terms(lda, 7)


#Clusting by Term Similarity
#remove a lot of the uninteresting or infrequent words.
dtmss <- removeSparseTerms(dtm, 0.15) # This makes a matrix that is only 15% empty space, maximum.
#dtmss

#Hierarchal Clustering
#First calculate distance between words & then cluster them according to similarity.

library(cluster)
d <- dist(t(dtmss), method="euclidian")
fit <- hclust(d=d, method="complete")   # for a different look try substituting: method="ward.D"
fit
```

```
##
## Call:
## hclust(d = d, method = "complete")
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 21
```

```
plot(fit, hang=-1)
```




```
#K-means clustering
#The k-means clustering method will attempt to cluster words into a specified number of
#groups (in this case 2), such that the sum of squared distances between individual words
#and one of the group centers. You can change the number of groups you seek by changing the number spec
library(fpc)
d <- dist(t(dtmss), method="euclidian")
kfit <- kmeans(d, 2)
clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0)
```




```
findAssocs(tdm, "korea", 0.2)
```

```
## $korea
##         south          deal    vietnamese     advocates       involve
##          0.49          0.35          0.33          0.32          0.32
## chinadailyusa          surv        repost   renegotiate         japan
##          0.30          0.30          0.29          0.26          0.23
##    japansouth        called  diplomatapac
##          0.22          0.21          0.20
```

```
findAssocs(tdm, "japan", 0.2)
```

```
## $japan
##     endlessly          deal    southkorea   renegotiate            ny
##          0.41          0.38          0.35          0.35          0.34
##     apologies        fought       veteran     agreement        called
##          0.30          0.29          0.29          0.28          0.28
##     advocates       involve            rt         pulls chinadailyusa
##          0.27          0.27          0.26          0.26          0.26
##          surv          with          amid         korea          from
##          0.26          0.25          0.24          0.23          0.21
##           row
##          0.21
```