Association Rule Mining and Twitter

Twitter API Setup

To access a Twitter API you will need to set up an account and receive a consumerKey, the comsumerSecret, the access_Token, and the access_Secret. A popular library and API: "twitteR".

Once you have your keys, you can set up the API.

```
requestURL='https://api.twitter.com/oauth/request_token'
accessURL='https://api.twitter.com/oauth/access_token'
authURL='https://api.twitter.com/oauth/authorize'

### NOTES: rtweet is another excellent option
## https://mkearney.github.io/blog/2017/06/01/intro-to-rtweet/
### https://rtweet.info/

### Install the needed packages...
#install.packages("twitteR")
#install.packages("ROAuth")
# install.packages("rtweet")
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
library(rtweet)
library(twitteR)
```

```
##
## Attaching package: 'twitteR'
```

```
## The following object is masked from 'package:rtweet':
##
##     lookup_statuses
```

```
library(ROAuth)
library(jsonlite)
```

```
##
## Attaching package: 'jsonlite'
```

```
## The following object is masked from 'package:rtweet':
##
##     flatten
```

```
#install.packages("streamR")
#library(streamR)
#install.packages("rjson")
library(rjson)
```

```
##
## Attaching package: 'rjson'

## The following objects are masked from 'package:jsonlite':
##
##     fromJSON, toJSON
```

```
#install.packages("tokenizers")
library(tokenizers)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- tidyv

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------------------- tidyverse_c
## x tidyr::expand()   masks Matrix::expand()
## x dplyr::filter()   masks stats::filter()
## x purrr::flatten()  masks jsonlite::flatten(), rtweet::flatten()
## x rjson::fromJSON() masks jsonlite::fromJSON()
## x dplyr::id()       masks twitteR::id()
## x dplyr::lag()      masks stats::lag()
## x dplyr::location() masks twitteR::location()
## x dplyr::recode()   masks arules::recode()
## x rjson::toJSON()   masks jsonlite::toJSON()
```

```
library(plyr)
```

```
## ------------------------------------------------------------------------------
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:purrr':
##
##     compact

## The following object is masked from 'package:twitteR':
##
##     id
```

```
library(dplyr)
library(ggplot2)
#install.packages("syuzhet")  ## sentiment analysis
library(syuzhet)
```

```
##
```

```
## Attaching package: 'syuzhet'
```

```
## The following object is masked from 'package:rtweet':
##
##     get_tokens
```

```
library(stringr)
#install.packages("arulesViz")
library(arulesViz)
```

```
## Loading required package: grid
```

Collecting Tweets

Next we will set up the API and search for a particular hash tag. We will store the tweets with the designated hash in a csv file for safe keeping. Here, we choose "#Trump" in hopes to get a 100 tweets easily.

```
##############  Using twittR ################################################################
setup_twitter_oauth(consumerKey,consumerSecret,access_Token,access_Secret)
```

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

```
Search<-twitteR::searchTwitter("#Trump",n=100,since="2018-09-09")
Search_DF <- twListToDF(Search)
TransactionTweetsFile = "Choc.csv"
Search_DF$text[1]
```

```
## [1] "RT @ricklevy67: #Turkey Advances on #Kobani in Latest Broken Promise\n#Erdogan told #Trump he w
```

```
## Start the file
Trans <- file(TransactionTweetsFile)
## Tokenize to words
Tokens<-tokenizers::tokenize_words(Search_DF$text[1],stopwords = stopwords::stopwords("en"),
          lowercase = TRUE,  strip_punct = TRUE, strip_numeric = TRUE,simplify = TRUE)
## Write squished tokens
cat(unlist(str_squish(Tokens)), "\n", file=Trans, sep=",")
close(Trans)

## Append remaining lists of tokens into file
## Recall - a list of tokens is the set of words from a Tweet
Trans <- file(TransactionTweetsFile, open = "a")
for(i in 2:nrow(Search_DF)){
  Tokens<-tokenize_words(Search_DF$text[i],stopwords = stopwords::stopwords("en"),
            lowercase = TRUE,  strip_punct = TRUE, simplify = TRUE)
  cat(unlist(str_squish(Tokens)), "\n", file=Trans, sep=",")
}
close(Trans)
```

Tweets as Transactions

In this section we will read in the tweets stored in the CSV file using the (Association Rule Mining) ARM library. Each tweet will be considered a basket of words. We can use ARM to determine associations of words in tweets.

```
######### Read in the tweet transactions
TweetTrans <- read.transactions(TransactionTweetsFile,
                                rm.duplicates = FALSE,
                                format = "basket",
                                sep=","
                                ## cols =
                                )
inspect(head(TweetTrans))
```

```
##     items
## [1] {advances,
##      attack,
##      broken,
##      erdogan,
##      important,
##      kobani,
##      ku,
##      latest,
##      promise,
##      rt,
##      symbolically,
##      told,
##      trump,
##      turkey}
## [2] {continue,
##      democrats,
##      hold,
##      house,
##      impeachmentinquiry,
##      pollsofpolitics,
##      r,
##      rt,
##      speakerpelosi,
##      vote}
## [3] {adamschiff,
##      bamboozled,
##      believing,
##      chanelrion,
##      cited,
##      comedians,
##      consulted,
##      evidence,
##      fake,
##      falsified,
##      rt,
##      russian,
##      testimony,
##      th}
## [4] {fullness,
##      jesus,
##      listen,
##      may,
##      potus,
##      president,
```

```
##      promting,
##      rest,
##      rt,
##      spirit,
##      trump,
##      veteransalways_}
## [5] {admitting,
##      considers,
##      day,
##      deeper,
##      dug,
##      end,
##      hims,
##      hole,
##      impeachment,
##      just,
##      mog7546,
##      mulvaney,
##      rt,
##      still,
##      strike,
##      trump,
##      two}
## [6] {adamschiff,
##      bamboozled,
##      believing,
##      chanelrion,
##      cited,
##      comedians,
##      consulted,
##      evidence,
##      fake,
##      falsified,
##      rt,
##      russian,
##      testimony,
##      th}
```

```r
## See the words that occur the most
Sample_Trans <- sample(TweetTrans, 50)
summary(Sample_Trans)
```

```
## transactions as itemMatrix in sparse format with
##  50 rows (elements/itemsets/transactions) and
##  788 columns (items) and a density of 0.01939086
##
## most frequent items:
##          rt       trump       https        t.co starknightz     (Other)
##          40          28          16          16          10         654
##
## element (itemset/transaction) length distribution:
## sizes
##  7  9 11 12 13 14 15 16 17 18 19 20 21 22
##  2  1  1  3  2  9  6  9 10  2  1  2  1  1
##
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.00   14.00   16.00   15.28   17.00   22.00
##
## includes extended item information - examples:
##       labels
## 1        000
## 2 0qwlqzxvjh
## 3         10
```

```
## Read the transactions data into a dataframe
TweetDF <- read.csv(TransactionTweetsFile, header = FALSE, sep = ",")
head(TweetDF)
```

```
##   V1              V2            V3        V4       V5     V6        V7
## 1 rt          turkey      advances    kobani   latest broken   promise
## 2 rt pollsofpolitics speakerpelosi      hold     vote  house democrats
## 3 rt      chanelrion    adamschiff falsified evidence  cited      fake
## 4 rt veteransalways_           may  fullness   spirit  jesus      rest
## 5 rt         mog7546        strike       two mulvaney   just       dug
## 6 rt      chanelrion    adamschiff falsified evidence  cited      fake
##            V8                V9       V10       V11          V12
## 1     erdogan               told     trump    attack symbolically
## 2    continue impeachmentinquiry         r
## 3   testimony          consulted   russian comedians   bamboozled
## 4       potus                may president     trump       listen
## 5 impeachment               hole    deeper admitting        trump
## 6   testimony          consulted   russian comedians   bamboozled
##         V13 V14   V15       V16  V17 V18
## 1 important  ku
## 2
## 3 believing  th
## 4  promting
## 5       end day still considers hims
## 6 believing  th
```

```
str(TweetDF)
```

```
## 'data.frame':    112 obs. of  18 variables:
##  $ V1 : Factor w/ 26 levels "afd","de","federación",..: 17 17 17 17 17 17 17 3 14 17 ...
##  $ V2 : Factor w/ 60 levels "","amjoyshow",..: 49 37 8 50 32 8 17 42 20 46 ...
##  $ V3 : Factor w/ 72 levels "","10","1000",..: 18 61 17 48 62 17 32 46 72 37 ...
##  $ V4 : Factor w/ 79 levels "","000","2020",..: 37 31 25 28 72 25 22 1 3 49 ...
##  $ V5 : Factor w/ 68 levels "","10","35","ahmed",..: 32 61 20 52 35 20 15 1 19 3 ...
##  $ V6 : Factor w/ 73 levels "","13","40","aac",..: 9 30 13 34 35 13 65 1 68 18 ...
##  $ V7 : Factor w/ 75 levels "","3","8bhwezazmc",..: 48 14 23 54 19 23 10 1 18 31 ...
##  $ V8 : Factor w/ 73 levels "","annewill",..: 23 13 63 50 33 63 21 1 8 47 ...
##  $ V9 : Factor w/ 75 levels "","2020","20th",..: 64 32 11 43 28 11 47 1 40 3 ...
##  $ V10: Factor w/ 65 levels "","000","125",..: 63 51 53 49 16 53 17 1 62 30 ...
##  $ V11: Factor w/ 66 levels "","000","15",..: 12 1 22 59 8 22 23 1 41 57 ...
##  $ V12: Factor w/ 62 levels "","0qwlqzxvjh",..: 49 1 10 31 54 10 16 1 24 39 ...
##  $ V13: Factor w/ 56 levels "","adams","amjoyshow",..: 27 1 7 43 20 7 6 1 47 50 ...
##  $ V14: Factor w/ 54 levels "","activated",..: 24 1 44 1 13 44 18 1 54 29 ...
##  $ V15: Factor w/ 49 levels "","1loa3ixzto",..: 1 1 1 1 36 1 11 1 1 43 ...
##  $ V16: Factor w/ 36 levels "","19tt8zziz0",..: 1 1 1 1 9 1 8 1 1 36 ...
##  $ V17: Factor w/ 30 levels "","2020","abd'ye",..: 1 1 1 1 14 1 9 1 1 1 ...
##  $ V18: Factor w/ 17 levels "","ausgerechnet",..: 1 1 1 1 1 1 9 1 1 1 ...
```

Cleaning the text data

Note that cleaning the text data is very important in text mining applications. Tweets are especially "messy". We will remove "rt", "http", etc and any other strings of no importance.

```
## Convert all columns to char
TweetDF<-TweetDF %>%
  mutate_all(as.character)
str(TweetDF)
```

```
## 'data.frame':    112 obs. of  18 variables:
##  $ V1 : chr  "rt" "rt" "rt" "rt" ...
##  $ V2 : chr  "turkey" "pollsofpolitics" "chanelrion" "veteransalways_" ...
##  $ V3 : chr  "advances" "speakerpelosi" "adamschiff" "may" ...
##  $ V4 : chr  "kobani" "hold" "falsified" "fullness" ...
##  $ V5 : chr  "latest" "vote" "evidence" "spirit" ...
##  $ V6 : chr  "broken" "house" "cited" "jesus" ...
##  $ V7 : chr  "promise" "democrats" "fake" "rest" ...
##  $ V8 : chr  "erdogan" "continue" "testimony" "potus" ...
##  $ V9 : chr  "told" "impeachmentinquiry" "consulted" "may" ...
##  $ V10: chr  "trump" "r" "russian" "president" ...
##  $ V11: chr  "attack" "" "comedians" "trump" ...
##  $ V12: chr  "symbolically" "" "bamboozled" "listen" ...
##  $ V13: chr  "important" "" "believing" "promting" ...
##  $ V14: chr  "ku" "" "th" "" ...
##  $ V15: chr  "" "" "" "" ...
##  $ V16: chr  "" "" "" "" ...
##  $ V17: chr  "" "" "" "" ...
##  $ V18: chr  "" "" "" "" ...
```

```
# We can now remove certain words
TweetDF[TweetDF == "t.co"] <- ""
TweetDF[TweetDF == "rt"] <- ""
TweetDF[TweetDF == "http"] <- ""
TweetDF[TweetDF == "https"] <- ""

## Clean with grepl - every row in each column
MyDF<-NULL
for (i in 1:ncol(TweetDF)){
  MyList=c() # each list is a column of logicals ...
  MyList=c(MyList,grepl("[[:digit:]]", TweetDF[[i]]))
  MyDF<-cbind(MyDF,MyList)  ## create a logical DF
  ## TRUE is when a cell has a word that contains digits
}
## For all TRUE, replace with blank
TweetDF[MyDF] <- ""
head(TweetDF,10)
```

```
##                 V1             V2            V3             V4       V5
## 1                          turkey      advances          kobani   latest
## 2                pollsofpolitics speakerpelosi            hold     vote
## 3                     chanelrion    adamschiff       falsified evidence
## 4                veteransalways_           may        fullness   spirit
## 5                                       strike             two mulvaney
## 6                     chanelrion    adamschiff       falsified evidence
## 7                   freddybernal     excelente       encuentro       de
```

```
## 8       federación            rusa     konstanti
## 9  realdonaldtrump          happen          win                      end
## 10                     starknightz     gopleader realdonaldtrump
##         V6        V7          V8                V9       V10       V11
## 1   broken   promise     erdogan              told     trump    attack
## 2    house democrats    continue impeachmentinquiry         r
## 3    cited      fake   testimony          consulted   russian comedians
## 4    jesus      rest       potus               may president     trump
## 5     just       dug impeachment              hole    deeper admitting
## 6    cited      fake   testimony          consulted   russian comedians
## 7  trabajo       con          el               pdte       del    comité
## 8
## 9       us      dogs        cats             living  together         m
## 10     day   history     october
##             V12       V13        V14   V15       V16  V17 V18
## 1  symbolically important         ku
## 2
## 3    bamboozled believing         th
## 4        listen  promting
## 5         trump       end        day still considers hims
## 6    bamboozled believing         th
## 7            de   asuntos exteriores   del   consejo   de  la
## 8
## 9
## 10   ojdhcvqxlo     trump      qanon             wwg
```

```
# Now we save the dataframe using the write table command
write.table(TweetDF, file = "UpdatedChocolate.csv", col.names = FALSE,
            row.names = FALSE, sep = ",")
TweetTrans <- read.transactions("UpdatedChocolate.csv", sep =",",
            format("basket"),  rm.duplicates = TRUE)
```

```
## distribution of transactions with duplicates:
## items
##  1  2  3  6
## 26  3  5  2
```

```
inspect(head(TweetTrans))
```

```
##     items
## [1] {advances,
##      attack,
##      broken,
##      erdogan,
##      important,
##      kobani,
##      ku,
##      latest,
##      promise,
##      symbolically,
##      told,
##      trump,
##      turkey}
## [2] {continue,
##      democrats,
```

```
##      hold,
##      house,
##      impeachmentinquiry,
##      pollsofpolitics,
##      r,
##      speakerpelosi,
##      vote}
## [3] {adamschiff,
##      bamboozled,
##      believing,
##      chanelrion,
##      cited,
##      comedians,
##      consulted,
##      evidence,
##      fake,
##      falsified,
##      russian,
##      testimony,
##      th}
## [4] {fullness,
##      jesus,
##      listen,
##      may,
##      potus,
##      president,
##      promting,
##      rest,
##      spirit,
##      trump,
##      veteransalways_}
## [5] {admitting,
##      considers,
##      day,
##      deeper,
##      dug,
##      end,
##      hims,
##      hole,
##      impeachment,
##      just,
##      mulvaney,
##      still,
##      strike,
##      trump,
##      two}
## [6] {adamschiff,
##      bamboozled,
##      believing,
##      chanelrion,
##      cited,
##      comedians,
##      consulted,
##      evidence,
```

```
##      fake,
##      falsified,
##      russian,
##      testimony,
##      th}
```

ARM

Next we will apply the apriori algorithm to find the associations including computing the support, confidence and lift. Read more on the arules library to tweak / tune the following code to achieve desired results.

```
# So that you do not have an enormous amount of rules, you can thresholds for
# support, confidence and lift ... also minlength for the rules.
TweetTrans_rules = arules::apriori(TweetTrans,
            parameter = list(support=.025, confidence=.75, minlen=3))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.75    0.1    1 none FALSE            TRUE       5   0.025      3
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 2
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[741 item(s), 112 transaction(s)] done [0.00s].
## sorting and recoding items ... [101 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.05s].
## writing ... [720392 rule(s)] done [0.24s].
## creating S4 object  ... done [0.57s].
```

```
inspect(head(TweetTrans_rules))
```

```
##     lhs                        rhs          support    confidence lift
## [1] {federación,konstanti} => {rusa}       0.02678571 1          37.33333
## [2] {federación,rusa}      => {konstanti}  0.02678571 1          37.33333
## [3] {konstanti,rusa}       => {federación} 0.02678571 1          37.33333
## [4] {release,via}          => {tax}        0.02678571 1          37.33333
## [5] {release,tax}          => {via}        0.02678571 1          37.33333
## [6] {tax,via}              => {release}    0.02678571 1          37.33333
##     count
## [1] 3
## [2] 3
## [3] 3
## [4] 3
## [5] 3
## [6] 3
```

```
## sorted
SortedRules_conf <- sort(TweetTrans_rules, by="confidence", decreasing=TRUE)
inspect(head(SortedRules_conf))
```

```
##     lhs                        rhs          support    confidence lift
## [1] {federación,konstanti} => {rusa}       0.02678571 1          37.33333
## [2] {federación,rusa}      => {konstanti}  0.02678571 1          37.33333
## [3] {konstanti,rusa}       => {federación} 0.02678571 1          37.33333
## [4] {release,via}          => {tax}        0.02678571 1          37.33333
## [5] {release,tax}          => {via}        0.02678571 1          37.33333
## [6] {tax,via}              => {release}    0.02678571 1          37.33333
##     count
## [1] 3
## [2] 3
## [3] 3
## [4] 3
## [5] 3
## [6] 3
```

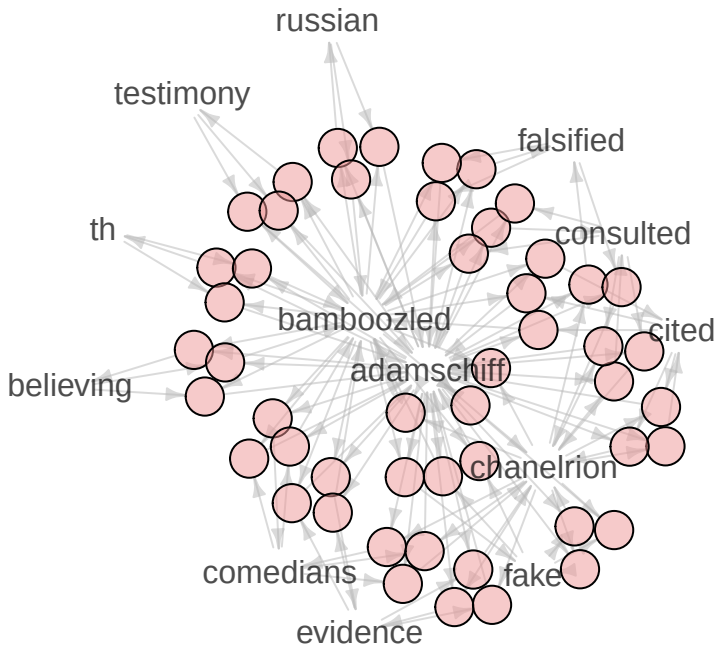
```
SortedRules_sup <- sort(TweetTrans_rules, by="support", decreasing=TRUE)
inspect(head(SortedRules_sup))
```

```
##     lhs                          rhs          support    confidence lift
## [1] {adamschiff,bamboozled} => {chanelrion} 0.08035714 1          12.44444
## [2] {adamschiff,chanelrion} => {bamboozled} 0.08035714 1          12.44444
## [3] {bamboozled,chanelrion} => {adamschiff} 0.08035714 1          12.44444
## [4] {adamschiff,bamboozled} => {cited}      0.08035714 1          12.44444
## [5] {adamschiff,cited}      => {bamboozled} 0.08035714 1          12.44444
## [6] {bamboozled,cited}      => {adamschiff} 0.08035714 1          12.44444
##     count
## [1] 9
## [2] 9
## [3] 9
## [4] 9
## [5] 9
## [6] 9
```

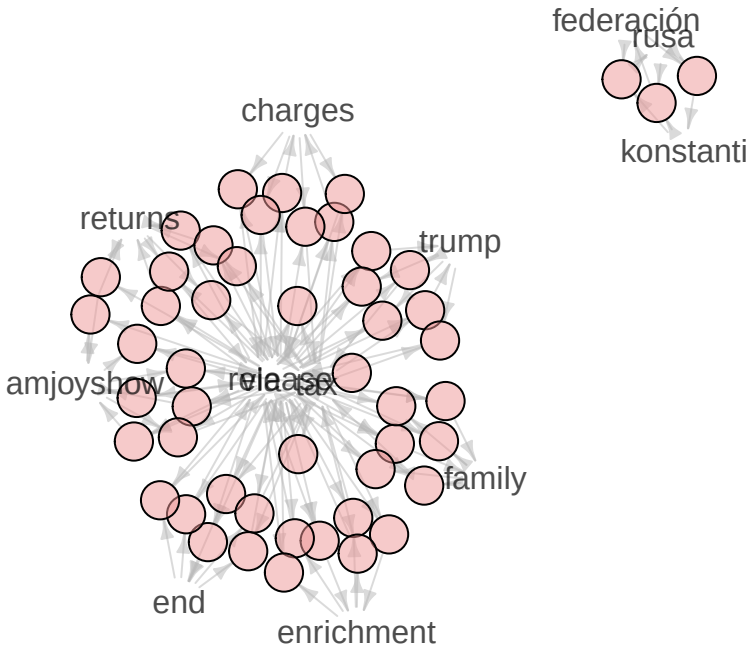
Displaying Results

The results will be displayed as an interactive graph.

```
plot (SortedRules_sup[1:50],method="graph",shading="confidence")
```




```
plot (SortedRules_conf[1:50],method="graph",shading="confidence")
```




```
#plot (SortedRules_sup[1:50],method="graph",interactive=TRUE,shading="confidence")
#plot (SortedRules_conf[1:50],method="graph",interactive=TRUE,shading="confidence")
```