Association Rule Mining and Twitter

Question 1

Connect to twitter and begin search!!

```
knitr::opts_chunk$set(echo = TRUE, warning=FALSE, results ='show',include=TRUE,messages=FALSE)

####### Twitter in R
#  Consumer API keys
#  Access token & access token secret

## I have created a text file that contains the
## consumerKey, the comsumerSecret, the access_Token, and the access_Secret
## They are comma seperated.
# Insert your consumerKey and consumerSecret below



consumerKey='SiMslBfTdWEimvLweRDTTrZVH'
consumerSecret='FoPYqK3uwpzutwE6G1RmQvPbRJ8RChFSLfIlgRAcFHjymKDzHh'
access_Token='1084502204038479872-v2czQaDlMt9ikoLnxhiQYk8Yb3f0RT'
access_Secret='U9ktzvd5rEwcK13mttsgwAujS0VxNPtJstxXcEE5znnid'
```

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

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

```
library(semPlot)
library(tm)
```

```
## Loading required package: NLP

##
## Attaching package: 'NLP'

## The following object is masked from 'package:ggplot2':
##
##     annotate

##
## Attaching package: 'tm'

## The following object is masked from 'package:arules':
##
##     inspect
```

```
library(wordcloud)
```

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

Collecting Tweets

Next we will set up the API and search for a particular string. .

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

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

```
Search<-twitteR::searchTwitter("nfl",n=90,since="2019-10-14")
Search_DF <- twListToDF(Search)
TransactionTweetsFile = "Choc.csv"
#Search_DF$text[1]
```

```
## 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)
tokenList = Tokens

## 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=",")
  tokenList <- c(tokenList,  unlist(str_squish(Tokens)))
}
close(Trans)
```

Question 2. Word Cloud

```
cor <- Corpus(VectorSource(tokenList))


tdm <- TermDocumentMatrix(cor)
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)

## NOTE:  d contains the words d$word AND frequencies d$freq

wordcloud(d$word,d$freq, colors=c("red","green","blue","orange","black","purple", "seagreen") , random.
```

Some words are not helpful for analysis in this context, so lets remove some of these so-called stopwords and re-create our wordcloud

```
tokenList[tokenList == "t.co"] <- ""
tokenList[tokenList == "rt"] <- ""
tokenList[tokenList == "http"] <- ""
tokenList[tokenList == "https"] <- ""
tokenList[tokenList == "sxrgihoe"] <- ""



cor <- Corpus(VectorSource(tokenList))


tdm <- TermDocumentMatrix(cor)
m <- as.matrix(tdm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)

## NOTE:  d contains the words d$word AND frequencies d$freq

wordcloud(d$word,d$freq, colors=c("red","green","blue","orange","black","purple", "seagreen") , random.
```

Question 3. Tweets as Transactions and identification of rules.

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(TweetTrans)
## See the words that occur the most
Sample_Trans <- sample(TweetTrans, 20)
#summary(Sample_Trans)

## 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     amhairdew    can      put  alll    refs    cover
## 2            rt kcchiefs_matt dustin colquitt  made history     last
```

```
## 3 patrickmahomes           nfl   best
## 4             rt    jagibbs_23  kenny golladay ranks   ninth       wr
## 5             rt  densmore_619   i've    never heard   sound     like
## 6             rt       espnnfl     10    years   ago   today tombrady
##         V8      V9    V10      V11      V12     V13        V14
## 1   madden  please asking   friend football    fans        nfl
## 2    night joining   jeff  feagles    shane lechler        one
## 3
## 4 position targets   nine      per     game   third        air
## 5     part haven't  since chargers   belong      sd nflcommish
## 6    threw     nfl record        5       td     one    quarter
##              V15     V16          V17  V18 V19 V20
## 1 nflofficiating
## 2          three players          nfl
## 3
## 4          yards   120.8          per game  th
## 5            nfl
## 6            one    best performances time via
```

```
#(str(TweetDF))
```

Re-Clean the text data if necessary ...

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':    93 obs. of  20 variables:
##  $ V1 : chr  "rt" "rt" "patrickmahomes" "rt" ...
##  $ V2 : chr  "amhairdew" "kcchiefs_matt" "nfl" "jagibbs_23" ...
##  $ V3 : chr  "can" "dustin" "best" "kenny" ...
##  $ V4 : chr  "put" "colquitt" "" "golladay" ...
##  $ V5 : chr  "alll" "made" "" "ranks" ...
##  $ V6 : chr  "refs" "history" "" "ninth" ...
##  $ V7 : chr  "cover" "last" "" "wr" ...
##  $ V8 : chr  "madden" "night" "" "position" ...
##  $ V9 : chr  "please" "joining" "" "targets" ...
##  $ V10: chr  "asking" "jeff" "" "nine" ...
##  $ V11: chr  "friend" "feagles" "" "per" ...
##  $ V12: chr  "football" "shane" "" "game" ...
##  $ V13: chr  "fans" "lechler" "" "third" ...
##  $ V14: chr  "nfl" "one" "" "air" ...
##  $ V15: chr  "nflofficiating" "three" "" "yards" ...
##  $ V16: chr  "" "players" "" "120.8" ...
##  $ V17: chr  "" "nfl" "" "per" ...
##  $ V18: chr  "" "" "" "game" ...
##  $ V19: chr  "" "" "" "th" ...
##  $ V20: chr  "" "" "" "" ...
```

```
## NULL
```

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

## 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       V6
## 1                    amhairdew     can        put       alll     refs
## 2                kcchiefs_matt  dustin   colquitt       made  history
## 3  patrickmahomes           nfl    best
## 4                               kenny   golladay      ranks    ninth
## 5                                i've      never      heard    sound
## 6                      espnnfl             years        ago    today
## 7                    jaysekulow   radio    planned parenthood employee
## 8        baldynfl        chiefs broncos        nfl    tackles      get
## 9                                colin kaepernick       best       qb
## 10        inemity           nfl  chiefs     what's difference     eric
##           V7       V8         V9    V10      V11         V12     V13
## 1      cover   madden     please asking   friend    football    fans
## 2       last    night    joining   jeff  feagles       shane lechler
## 3
## 4         wr position    targets   nine      per        game   third
## 5       like     part    haven't  since chargers      belong      sd
## 6   tombrady    threw        nfl record                  td     one
## 7      jokes  selling       baby   body    parts lamborghini
## 8  destroyed    royce    freeman slides    makes        sure  nobody
## 9                nfl      teams  right      now        sign
## 10      reid  protest kaepernick
##           V14            V15     V16          V17  V18 V19 V20
## 1         nfl nflofficiating
## 2         one          three players          nfl
## 3
## 4         air          yards                 per game  th
## 5  nflcommish            nfl
## 6     quarter            one    best performances time via
## 7
## 8        tape
## 9
## 10
```

```r
# 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
## 9 5 1
```

```r
#inspect(TweetTrans)
```

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.

```r
# 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=0.05, confidence=.45, minlen=3))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.45    0.1    1 none FALSE            TRUE       5    0.05      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: 4
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[583 item(s), 93 transaction(s)] done [0.00s].
## sorting and recoding items ... [26 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 [0.02s].
## writing ... [223499 rule(s)] done [0.10s].
## creating S4 object  ... done [0.27s].
```

```r
arules::inspect(head(TweetTrans_rules, 10))
```

```
##      lhs                       rhs            support    confidence
## [1]  {anniversary,worst}    => {history}      0.05376344 1.0000000
## [2]  {anniversary,history}  => {worst}        0.05376344 1.0000000
## [3]  {history,worst}        => {anniversary}  0.05376344 0.8333333
## [4]  {anniversary,worst}    => {nfl}          0.05376344 1.0000000
## [5]  {anniversary,nfl}      => {worst}        0.05376344 1.0000000
## [6]  {nfl,worst}            => {anniversary}  0.05376344 0.8333333
```

```
## [7]  {anniversary,history} => {nfl}          0.05376344 1.0000000
## [8]  {anniversary,nfl}     => {history}      0.05376344 1.0000000
## [9]  {history,nfl}         => {anniversary}  0.05376344 0.6250000
## [10] {quarter,threw}       => {performances} 0.06451613 1.0000000
##      lift     count
## [1]  11.62500 5
## [2]  13.28571 5
## [3]  15.50000 5
## [4]   1.55000 5
## [5]  13.28571 5
## [6]  15.50000 5
## [7]   1.55000 5
## [8]  11.62500 5
## [9]  11.62500 5
## [10] 15.50000 6
```

```
## sorted
SortedRules_conf <- sort(TweetTrans_rules, by="confidence", decreasing=TRUE)
arules::inspect(head(SortedRules_conf, 10))
```

```
##      lhs                       rhs            support    confidence
## [1]  {anniversary,worst}    => {history}      0.05376344 1
## [2]  {anniversary,history}  => {worst}        0.05376344 1
## [3]  {anniversary,worst}    => {nfl}          0.05376344 1
## [4]  {anniversary,nfl}      => {worst}        0.05376344 1
## [5]  {anniversary,history}  => {nfl}          0.05376344 1
## [6]  {anniversary,nfl}      => {history}      0.05376344 1
## [7]  {quarter,threw}        => {performances} 0.06451613 1
## [8]  {performances,threw}   => {quarter}      0.06451613 1
## [9]  {performances,quarter} => {threw}        0.06451613 1
## [10] {quarter,threw}        => {espnnfl}      0.06451613 1
##      lift     count
## [1]  11.62500 5
## [2]  13.28571 5
## [3]   1.55000 5
## [4]  13.28571 5
## [5]   1.55000 5
## [6]  11.62500 5
## [7]  15.50000 6
## [8]  15.50000 6
## [9]  15.50000 6
## [10] 15.50000 6
```

```
SortedRules_sup <- sort(TweetTrans_rules, by="support", decreasing=TRUE)
arules::inspect(head(SortedRules_sup, 10))
```

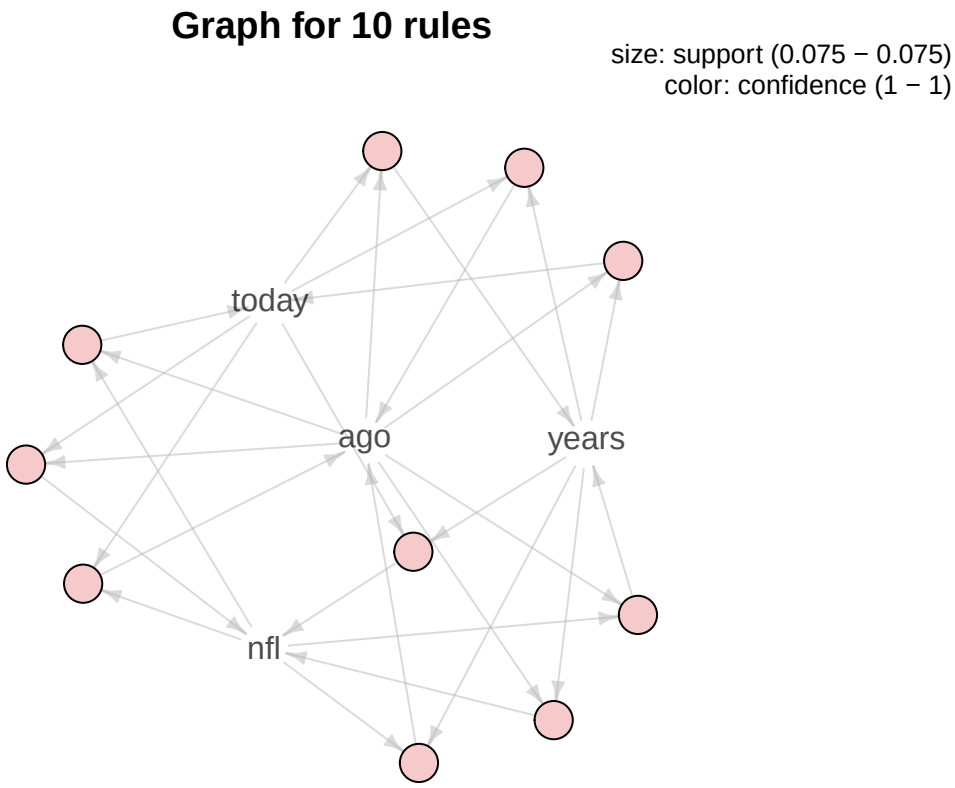
```
##      lhs              rhs     support    confidence lift     count
## [1]  {ago,years}   => {today} 0.07526882 1          13.28571 7
## [2]  {ago,today}   => {years} 0.07526882 1          13.28571 7
## [3]  {today,years} => {ago}   0.07526882 1          13.28571 7
## [4]  {ago,years}   => {nfl}   0.07526882 1           1.55000 7
## [5]  {ago,nfl}     => {years} 0.07526882 1          13.28571 7
## [6]  {nfl,years}   => {ago}   0.07526882 1          13.28571 7
## [7]  {ago,today}   => {nfl}   0.07526882 1           1.55000 7
## [8]  {ago,nfl}     => {today} 0.07526882 1          13.28571 7
```

```
## [9]  {nfl,today}   => {ago}   0.07526882 1          13.28571 7
## [10] {today,years} => {nfl}   0.07526882 1           1.55000 7
```

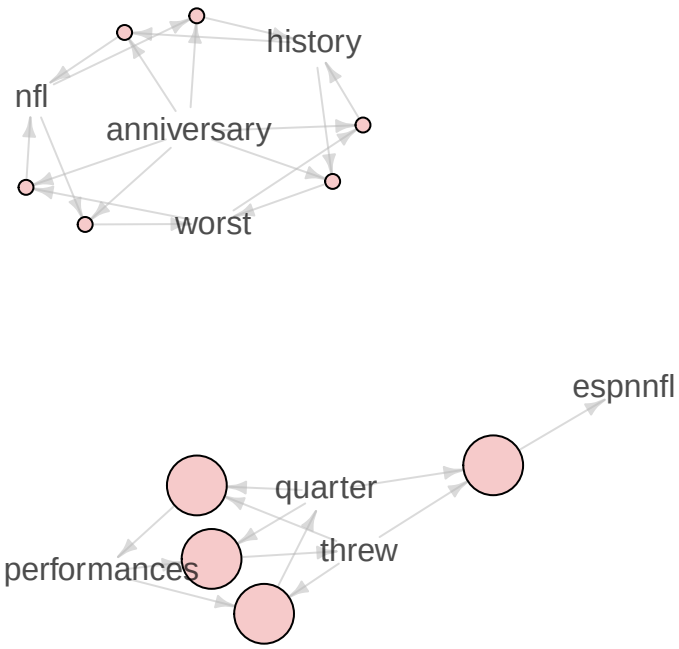
Question 4. Displaying Results

The results will be displayed as a graph.

```
plot (head(SortedRules_sup,n=10),method="graph",shading="confidence")
```




```
plot (head(SortedRules_conf, n=10),method="graph",shading="confidence")
```


Graph for 10 rules
size: support (0.054 − 0.065)
color: confidence (1 − 1)
history
nfl
anniversary
worst
espnnfl
quarter
threw
performances