ARM_RCode

Rae Zhang

10/26/2021

```
library("knitr")
library(rlang)
library(usethis)
library(devtools)
#install.packages("base64enc")
library(base64enc)
#install.packages("RCurl")
library(RCurl)
library(httr)
library(twitteR)
library(ROAuth)
library(networkD3)
library(arules)
library(rtweet)
library(jsonlite)
library(streamR)
library(rjson)
library(tokenizers)
library(tidyverse)
library(plyr)
library(dplyr)
library(ggplot2)
library(syuzhet)
library(stringr)
library(arulesViz)
library(igraph)
library(httpuv)
library(openssl)
library(rtweet)
### Since the twitteR package is in process of being deprecated, rtweet packa
ge is being used below
#https://www.rdocumentation.org/packages/rtweet/versions/0.7.0/topics/search_
tweets
# install.packages("devtools")
# devtools::install_github("mkearney/rtweet")
# auth_setup_default()
TwitterCodesFile="~/Documents/Georgetown/ANLY501/twitter_keys.txt"
tokens<-read.csv(TwitterCodesFile, header=T, sep=",")
consumerKey=as.character(tokens$API_key)
consumerSecret=as.character(tokens$API_key_secret)
access_Token=as.character(tokens$Access_Token)
access_Secret=as.character(tokens$Access_Token_Secret)
```

```
token <- create_token(
  app = "ANLY501_APP",
  consumerKey,
  consumerSecret,
  access_Token,
  access_Secret)

Search_DF<-search_tweets(
  "organic food",
  n = 10000, lang = "en", type = "mixed")

Search_DF1<-search_tweets(
  q = "#organicfood",
  n = 18000, lang = "en")

TransactionTweetsFile = "TweetResults.csv"
(Search_DF$text[1])
```

```
## [1] "#IFFCO is also strengthening its presence in #Organics. It’s ventures
 @sifco_ltd in organic food processing and @AquagriLimited in Agri Inputs &am
p; crop protection along with Urban Gardening @iffcourbangrdns showcasing at
Organics Exhibition @BioFachVivaness India. @nstomar @AgriGoI https://t.co/Oe
HIqd7ddY"
```

```
TransactionTweetsFile = "TweetResults1.csv"
(Search_DF1$text[1])
```

```
## [1] "Straight from the farm…👩‍🌾\nFood that doesn’t do any harm!💪 \nKeep l
ife #Simple and #Soulful \n❤️\ufe0f🥬🥦🥔🥕🫚🫑🍅🥕🥜❤️\ufe0f\n.\n.\n.\n.
\n.\n#SwasthRahoMastRaho #OrganicFood #healthylifestyle #cleaneating #eatrigh
t #growyourown #FarmToMouth https://t.co/fRwPphvUub"
```

```
Trans <- file(TransactionTweetsFile)

## Tokenize to words
Tokens<-tokenizers::tokenize_words(
  Search_DF1$text[1],stopwords = stopwords::stopwords("en"),
  lowercase = T,  strip_punct = T, strip_numeric = T,
  simplify = T)
```

Write tokens

```
cat(unlist(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_DF1)){
```

```
  Tokens<-tokenize_words(Search_DF1$text[i],
                         stopwords = stopwords::stopwords("en"),
                         lowercase = T,
                         strip_punct = T,
                         simplify = T)

  cat(unlist(Tokens), "\n", file=Trans, sep=",")
  cat(unlist(Tokens))
}

close(Trans)
```

Read in the tweet transactions

Read the transactions data into a dataframe

```
TweetDF <- read.csv(TransactionTweetsFile,
                    header = FALSE, sep = ",")
head(TweetDF)
```

```
##         V1            V2         V3      V4        V5         V6     V7
## 1 straight          farm       food doesn’t      harm       keep   life
## 2 straight          farm       food doesn’t      harm       keep   life
## 3    velma           mct    coconut     oil      mcts     appear induce
## 4   pulses       organic nutritious harmful chemicals pesticides   used
## 5    happy international      chefs     day     chefs     making  lives
## 6     give         meals      tadka organic  tattva's        red chilly
##              V8            V9                V10            V11
## 1        simple       soulful swasthrahomastraho     organicfood
## 2        simple       soulful swasthrahomastraho     organicfood
## 3 thermogenesis          heat         generation            body
## 4   cultivating         makes                fit         consume
## 5        better           one               dish            time
## 6        powder organictattva        organicfood organicproducts
##                  V12         V13                V14         V15
V16
## 1    healthylifestyle cleaneating           eatright growyourown farmtomo
uth
## 2    healthylifestyle cleaneating           eatright growyourown farmtomo
uth
## 3             helping     dieters               burn         fat      red
uce
## 4           healthier    compared      conventionally       grown      pul
ses
## 5 internationchefsday organicfood healthyanddelicious    chefsday       ht
tps
## 6             organic healthyfood       healthyliving      spices       ht
tps
##       V17             V18        V19       V20           V21         V22
## 1   https            t.co frwpphvuub
## 2   https            t.co frwpphvuub
```

```
## 3  weight helthylifestyle   ketodiet paleodiet     veganfood organicfood
## 4 organic          tattva     offers      just organictattva organicfood
## 5    t.co      hhd2ydqasr
## 6    t.co      a9lpucfvhv
##          V23        V24        V25 V26 V27    V28   V29  V30        V31 V3
2
## 1

## 2

## 3 weightloss metabolism   coconuts oil usa export https t.co 0s3lya7zfd

## 4      https       t.co 1newg65kfe

## 5

## 6

(str(TweetDF))

## 'data.frame':    462 obs. of  32 variables:
##  $ V1 : chr  "straight" "straight" "velma" "pulses" ...
##  $ V2 : chr  "farm" "farm" "mct" "organic" ...
##  $ V3 : chr  "food" "food" "coconut" "nutritious" ...
##  $ V4 : chr  "doesn’t" "doesn’t" "oil" "harmful" ...
##  $ V5 : chr  "harm" "harm" "mcts" "chemicals" ...
##  $ V6 : chr  "keep" "keep" "appear" "pesticides" ...
##  $ V7 : chr  "life" "life" "induce" "used" ...
##  $ V8 : chr  "simple" "simple" "thermogenesis" "cultivating" ...
##  $ V9 : chr  "soulful" "soulful" "heat" "makes" ...
##  $ V10: chr  "swasthrahomastraho" "swasthrahomastraho" "generation" "fit"
...
##  $ V11: chr  "organicfood" "organicfood" "body" "consume" ...
##  $ V12: chr  "healthylifestyle" "healthylifestyle" "helping" "healthier" .
..
##  $ V13: chr  "cleaneating" "cleaneating" "dieters" "compared" ...
##  $ V14: chr  "eatright" "eatright" "burn" "conventionally" ...
##  $ V15: chr  "growyourown" "growyourown" "fat" "grown" ...
##  $ V16: chr  "farmtomouth" "farmtomouth" "reduce" "pulses" ...
##  $ V17: chr  "https" "https" "weight" "organic" ...
##  $ V18: chr  "t.co" "t.co" "helthylifestyle" "tattva" ...
##  $ V19: chr  "frwpphvuub" "frwpphvuub" "ketodiet" "offers" ...
##  $ V20: chr  "" "" "paleodiet" "just" ...
##  $ V21: chr  "" "" "veganfood" "organictattva" ...
##  $ V22: chr  "" "" "organicfood" "organicfood" ...
##  $ V23: chr  "" "" "weightloss" "https" ...
##  $ V24: chr  "" "" "metabolism" "t.co" ...
##  $ V25: chr  "" "" "coconuts" "1newg65kfe" ...
##  $ V26: chr  "" "" "oil" "" ...
##  $ V27: chr  "" "" "usa" "" ...
```

```
##  $ V28: chr  "" "" "export" "" ...
##  $ V29: chr  "" "" "https" "" ...
##  $ V30: chr  "" "" "t.co" "" ...
##  $ V31: chr  "" "" "0s3lya7zfd" "" ...
##  $ V32: chr  "" "" "" "" ...

## NULL
```

Convert all columns to char

```
TweetDF<-TweetDF %>%
  mutate_all(as.character)
(str(TweetDF))

## 'data.frame':    462 obs. of  32 variables:
##  $ V1 : chr  "straight" "straight" "velma" "pulses" ...
##  $ V2 : chr  "farm" "farm" "mct" "organic" ...
##  $ V3 : chr  "food" "food" "coconut" "nutritious" ...
##  $ V4 : chr  "doesn't" "doesn't" "oil" "harmful" ...
##  $ V5 : chr  "harm" "harm" "mcts" "chemicals" ...
##  $ V6 : chr  "keep" "keep" "appear" "pesticides" ...
##  $ V7 : chr  "life" "life" "induce" "used" ...
##  $ V8 : chr  "simple" "simple" "thermogenesis" "cultivating" ...
##  $ V9 : chr  "soulful" "soulful" "heat" "makes" ...
##  $ V10: chr  "swasthrahomastraho" "swasthrahomastraho" "generation" "fit"
...
##  $ V11: chr  "organicfood" "organicfood" "body" "consume" ...
##  $ V12: chr  "healthylifestyle" "healthylifestyle" "helping" "healthier" .
..
##  $ V13: chr  "cleaneating" "cleaneating" "dieters" "compared" ...
##  $ V14: chr  "eatright" "eatright" "burn" "conventionally" ...
##  $ V15: chr  "growyourown" "growyourown" "fat" "grown" ...
##  $ V16: chr  "farmtomouth" "farmtomouth" "reduce" "pulses" ...
##  $ V17: chr  "https" "https" "weight" "organic" ...
##  $ V18: chr  "t.co" "t.co" "helthylifestyle" "tattva" ...
##  $ V19: chr  "frwpphvuub" "frwpphvuub" "ketodiet" "offers" ...
##  $ V20: chr  "" "" "paleodiet" "just" ...
##  $ V21: chr  "" "" "veganfood" "organictattva" ...
##  $ V22: chr  "" "" "organicfood" "organicfood" ...
##  $ V23: chr  "" "" "weightloss" "https" ...
##  $ V24: chr  "" "" "metabolism" "t.co" ...
##  $ V25: chr  "" "" "coconuts" "1newg65kfe" ...
##  $ V26: chr  "" "" "oil" "" ...
##  $ V27: chr  "" "" "usa" "" ...
##  $ V28: chr  "" "" "export" "" ...
##  $ V29: chr  "" "" "https" "" ...
##  $ V30: chr  "" "" "t.co" "" ...
##  $ V31: chr  "" "" "0s3lya7zfd" "" ...
##  $ V32: chr  "" "" "" "" ...

## NULL
```

Remove certain words

```
TweetDF[TweetDF == "t.co"] <- ""
TweetDF[TweetDF == "rt"] <- ""
TweetDF[TweetDF == "http"] <- ""
TweetDF[TweetDF == "https"] <- ""
TweetDF[TweetDF == "it’s"] <- ""
TweetDF[TweetDF == "really"] <- ""
TweetDF[TweetDF == "literally"] <- ""
TweetDF[TweetDF == "literal"] <- ""
TweetDF[TweetDF == "actually"] <- ""
```

Clean with grepl - every row in each column

```
MyDF<-NULL
MyDF2<-NULL
for (i in 1:ncol(TweetDF)){
  MyList=c()
  MyList2=c() # each list is a column of logicals ...
  MyList=c(MyList,grepl("[[:digit:]]", TweetDF[[i]]))
  MyDF<-cbind(MyDF,MyList)  ## create a logical DF
  MyList2=c(MyList2,(nchar(TweetDF[[i]])<4 | nchar(TweetDF[[i]])>10))
  MyDF2<-cbind(MyDF2,MyList2)
}
```

For all TRUE, replace with blank

```
TweetDF[MyDF] <- ""
TweetDF[MyDF2] <- ""
(head(TweetDF,10))
```

```
##          V1         V2         V3      V4        V5         V6       V7
  V8
## 1  straight       farm       food doesn’t      harm       keep     life si
mple
## 2  straight       farm       food doesn’t      harm       keep     life si
mple
## 3     velma               coconut             mcts     appear   induce

## 4    pulses    organic nutritious harmful chemicals pesticides     used

## 5     happy                 chefs            chefs     making    lives be
tter
## 6      give      meals      tadka organic  tattva's              chilly po
wder
## 7  favorite  breakfast  healthier  switch    better              living sw
itch
## 8     built              friendly  system    always       fond  promote na
ture
## 9    master       plan                   standards

## 10   mivida pakistan's              plots      open    booking offering
```

```
##         V9        V10     V11       V12      V13      V14      V15      V1
6
## 1  soulful                                    eatright
## 2  soulful                                    eatright
## 3     heat generation    body   helping  dieters     burn            reduc
e
## 4    makes            consume healthier compared             grown   pulse
s
## 5                dish    time                            chefsday
## 6                               organic                    spices
## 7  organic     tattva                                    organic
## 8   mivida   pakistan  nature   partner   mivida
## 9    smart       join  mivida    mivida                          propert
y
## 10   marla              kanal     plots             years     easy
##         V17      V18        V19       V20        V21 V22        V23
 V24
## 1                   frwpphvuub
## 2                   frwpphvuub
## 3    weight            ketodiet paleodiet  veganfood     weightloss metabo
lism
## 4   organic   tattva     offers      just
## 5
## 6
## 7
## 8  property               trees  greenery                healthyair     fu
ture
## 9           greenery                     healthyair                    fu
ture
## 10   mivida                                 property          trees   gree
nery
##         V25 V26        V27    V28 V29 V30 V31 V32
## 1
## 2
## 3  coconuts                export
```

```
## 4
## 5
## 6
## 7
## 8
## 9
## 10              healthyair future
```

Save the dataframe using the write table command

```
write.table(TweetDF, file = "UpdatedTweetFile.csv", col.names = FALSE,
            row.names = FALSE, sep = ",")
TweetTrans <- read.transactions("UpdatedTweetFile.csv", sep =",",
                                format("basket"),  rm.duplicates = TRUE)

## distribution of transactions with duplicates:
## items
##   1   2   3   4   5   6   7  12
## 115  69  37  24   5   1   1   1
```

Create the Rules - Relationships

```
TweetTrans_rules = arules::apriori(TweetTrans,
                                   parameter = list(support=0.02, conf=0.5, m
axlen=2))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5    0.02      1
##  maxlen target  ext
##       2  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 9
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[1633 item(s), 462 transaction(s)] done [0.00s].
## sorting and recoding items ... [102 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2

## Warning in arules::apriori(TweetTrans, parameter = list(support = 0.02, :
Mining
## stopped (maxlen reached). Only patterns up to a length of 2 returned!
```

```
##  done [0.00s].
## writing ... [511 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].

#appearance = list (default="lhs",rhs="milk")
inspect(TweetTrans_rules[1:15])

##      lhs             rhs          support    confidence coverage   lift
## [1]  {wellness}   => {organic}    0.02164502 1.0000000  0.02164502  2.0810
81
## [2]  {eatright}   => {doesn't}    0.02380952 1.0000000  0.02380952 42.0000
00
## [3]  {doesn't}    => {eatright}   0.02380952 1.0000000  0.02380952 42.0000
00
## [4]  {eatright}   => {frwpphvuub} 0.02380952 1.0000000  0.02380952 42.0000
00
## [5]  {frwpphvuub} => {eatright}   0.02380952 1.0000000  0.02380952 42.0000
00
## [6]  {eatright}   => {straight}   0.02380952 1.0000000  0.02380952 42.0000
00
## [7]  {straight}   => {eatright}   0.02380952 1.0000000  0.02380952 42.0000
00
## [8]  {eatright}   => {soulful}    0.02380952 1.0000000  0.02380952 42.0000
00
## [9]  {soulful}    => {eatright}   0.02380952 1.0000000  0.02380952 42.0000
00
## [10] {eatright}   => {keep}       0.02380952 1.0000000  0.02380952 38.5000
00
## [11] {keep}       => {eatright}   0.02380952 0.9166667  0.02597403 38.5000
00
## [12] {eatright}   => {harm}       0.02380952 1.0000000  0.02380952 35.5384
62
## [13] {harm}       => {eatright}   0.02380952 0.8461538  0.02813853 35.5384
62
## [14] {eatright}   => {simple}     0.02380952 1.0000000  0.02380952 33.0000
00
## [15] {simple}     => {eatright}   0.02380952 0.7857143  0.03030303 33.0000
00
##      count
## [1]  10
## [2]  11
## [3]  11
## [4]  11
## [5]  11
## [6]  11
## [7]  11
## [8]  11
## [9]  11
## [10] 11
```

```
## [11] 11
## [12] 11
## [13] 11
## [14] 11
## [15] 11
```

Sort by Conf

```
SortedRules_conf <- sort(TweetTrans_rules, by="confidence", decreasing=TRUE)
inspect(SortedRules_conf[1:15])
```

```
##      lhs             rhs          support    confidence coverage   lift
## [1]  {wellness}   => {organic}    0.02164502 1          0.02164502  2.0810
81
## [2]  {eatright}   => {doesn't}    0.02380952 1          0.02380952 42.0000
00
## [3]  {doesn't}    => {eatright}   0.02380952 1          0.02380952 42.0000
00
## [4]  {eatright}   => {frwpphvuub} 0.02380952 1          0.02380952 42.0000
00
## [5]  {frwpphvuub} => {eatright}   0.02380952 1          0.02380952 42.0000
00
## [6]  {eatright}   => {straight}   0.02380952 1          0.02380952 42.0000
00
## [7]  {straight}   => {eatright}   0.02380952 1          0.02380952 42.0000
00
## [8]  {eatright}   => {soulful}    0.02380952 1          0.02380952 42.0000
00
## [9]  {soulful}    => {eatright}   0.02380952 1          0.02380952 42.0000
00
## [10] {eatright}   => {keep}       0.02380952 1          0.02380952 38.5000
00
## [11] {eatright}   => {harm}       0.02380952 1          0.02380952 35.5384
62
## [12] {eatright}   => {simple}     0.02380952 1          0.02380952 33.0000
00
## [13] {eatright}   => {life}       0.02380952 1          0.02380952 27.1764
71
## [14] {eatright}   => {farm}       0.02380952 1          0.02380952 14.9032
26
## [15] {eatright}   => {food}       0.02380952 1          0.02380952  3.3722
63
##      count
## [1]  10
## [2]  11
## [3]  11
## [4]  11
## [5]  11
## [6]  11
## [7]  11
```

```
## [8]  11
## [9]  11
## [10] 11
## [11] 11
## [12] 11
## [13] 11
## [14] 11
## [15] 11
```

Sort by Sup

```
SortedRules_sup <- sort(TweetTrans_rules, by="support", decreasing=TRUE)
inspect(SortedRules_sup[1:15])
```

```
##      lhs             rhs          support    confidence coverage   lift
## [1]  {food}       => {organic}    0.18614719 0.6277372  0.29653680  1.3063
72
## [2]  {products}   => {organic}    0.09523810 0.8461538  0.11255411  1.7609
15
## [3]  {health}     => {organic}    0.07142857 0.6734694  0.10606061  1.4015
44
## [4]  {nirvana}    => {organic}    0.06926407 1.0000000  0.06926407  2.0810
81
## [5]  {products}   => {food}       0.06277056 0.5576923  0.11255411  1.8806
85
## [6]  {mukteshwar} => {organic}    0.05627706 1.0000000  0.05627706  2.0810
81
## [7]  {store}      => {organic}    0.05627706 1.0000000  0.05627706  2.0810
81
## [8]  {nirvana}    => {food}       0.05627706 0.8125000  0.06926407  2.7399
64
## [9]  {health}     => {food}       0.05411255 0.5102041  0.10606061  1.7205
42
## [10] {mukteshwar} => {nirvana}    0.04978355 0.8846154  0.05627706 12.7716
35
## [11] {nirvana}    => {mukteshwar} 0.04978355 0.7187500  0.06926407 12.7716
35
## [12] {india}      => {food}       0.04978355 0.7419355  0.06709957  2.5020
01
## [13] {india}      => {organic}    0.04978355 0.7419355  0.06709957  1.5440
28
## [14] {nirvana}    => {products}   0.04545455 0.6562500  0.06926407  5.8305
29
## [15] {store}      => {nirvana}    0.04329004 0.7692308  0.05627706 11.1057
69
##      count
## [1]  86
## [2]  44
## [3]  33
## [4]  32
```

```
## [5]  29
## [6]  26
## [7]  26
## [8]  26
## [9]  25
## [10] 23
## [11] 23
## [12] 23
## [13] 23
## [14] 21
## [15] 20
```

Sort by Lift

```
SortedRules_lift <- sort(TweetTrans_rules, by="lift", decreasing=TRUE)
inspect(SortedRules_lift[1:15])
```

```
##      lhs             rhs          support    confidence coverage   lift co
unt
## [1]  {eatright}   => {doesn't}    0.02380952 1          0.02380952 42   11

## [2]  {doesn't}    => {eatright}   0.02380952 1          0.02380952 42   11

## [3]  {eatright}   => {frwpphvuub} 0.02380952 1          0.02380952 42   11

## [4]  {frwpphvuub} => {eatright}   0.02380952 1          0.02380952 42   11

## [5]  {eatright}   => {straight}   0.02380952 1          0.02380952 42   11

## [6]  {straight}   => {eatright}   0.02380952 1          0.02380952 42   11

## [7]  {eatright}   => {soulful}    0.02380952 1          0.02380952 42   11

## [8]  {soulful}    => {eatright}   0.02380952 1          0.02380952 42   11

## [9]  {doesn't}    => {frwpphvuub} 0.02380952 1          0.02380952 42   11

## [10] {frwpphvuub} => {doesn't}    0.02380952 1          0.02380952 42   11

## [11] {doesn't}    => {straight}   0.02380952 1          0.02380952 42   11

## [12] {straight}   => {doesn't}    0.02380952 1          0.02380952 42   11

## [13] {doesn't}    => {soulful}    0.02380952 1          0.02380952 42   11

## [14] {soulful}    => {doesn't}    0.02380952 1          0.02380952 42   11

## [15] {frwpphvuub} => {straight}   0.02380952 1          0.02380952 42   11
```

```
TweetTrans_rules<-SortedRules_lift[1:50]
inspect(TweetTrans_rules)
```

```
##      lhs             rhs          support    confidence coverage   lift co
unt
## [1]  {eatright}   => {doesn't}    0.02380952 1.0000000  0.02380952 42.0 11

## [2]  {doesn't}    => {eatright}   0.02380952 1.0000000  0.02380952 42.0 11

## [3]  {eatright}   => {frwpphvuub} 0.02380952 1.0000000  0.02380952 42.0 11

## [4]  {frwpphvuub} => {eatright}   0.02380952 1.0000000  0.02380952 42.0 11

## [5]  {eatright}   => {straight}   0.02380952 1.0000000  0.02380952 42.0 11

## [6]  {straight}   => {eatright}   0.02380952 1.0000000  0.02380952 42.0 11

## [7]  {eatright}   => {soulful}    0.02380952 1.0000000  0.02380952 42.0 11

## [8]  {soulful}    => {eatright}   0.02380952 1.0000000  0.02380952 42.0 11

## [9]  {doesn't}    => {frwpphvuub} 0.02380952 1.0000000  0.02380952 42.0 11

## [10] {frwpphvuub} => {doesn't}    0.02380952 1.0000000  0.02380952 42.0 11

## [11] {doesn't}    => {straight}   0.02380952 1.0000000  0.02380952 42.0 11

## [12] {straight}   => {doesn't}    0.02380952 1.0000000  0.02380952 42.0 11

## [13] {doesn't}    => {soulful}    0.02380952 1.0000000  0.02380952 42.0 11

## [14] {soulful}    => {doesn't}    0.02380952 1.0000000  0.02380952 42.0 11

## [15] {frwpphvuub} => {straight}   0.02380952 1.0000000  0.02380952 42.0 11

## [16] {straight}   => {frwpphvuub} 0.02380952 1.0000000  0.02380952 42.0 11

## [17] {frwpphvuub} => {soulful}    0.02380952 1.0000000  0.02380952 42.0 11

## [18] {soulful}    => {frwpphvuub} 0.02380952 1.0000000  0.02380952 42.0 11

## [19] {straight}   => {soulful}    0.02380952 1.0000000  0.02380952 42.0 11

## [20] {soulful}    => {straight}   0.02380952 1.0000000  0.02380952 42.0 11

## [21] {eatright}   => {keep}       0.02380952 1.0000000  0.02380952 38.5 11

## [22] {keep}       => {eatright}   0.02380952 0.9166667  0.02597403 38.5 11
```

```
## [23] {doesn’t}    => {keep}       0.02380952 1.0000000  0.02380952 38.5 11
## [24] {keep}       => {doesn’t}    0.02380952 0.9166667  0.02597403 38.5 11
## [25] {frwpphvuub} => {keep}       0.02380952 1.0000000  0.02380952 38.5 11
## [26] {keep}       => {frwpphvuub} 0.02380952 0.9166667  0.02597403 38.5 11
## [27] {straight}   => {keep}       0.02380952 1.0000000  0.02380952 38.5 11
## [28] {keep}       => {straight}   0.02380952 0.9166667  0.02597403 38.5 11
## [29] {soulful}    => {keep}       0.02380952 1.0000000  0.02380952 38.5 11
## [30] {keep}       => {soulful}    0.02380952 0.9166667  0.02597403 38.5 11
## [31] {turin}      => {social}     0.02597403 1.0000000  0.02597403 38.5 12
## [32] {social}     => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
## [33] {turin}      => {practices}  0.02597403 1.0000000  0.02597403 38.5 12
## [34] {practices}  => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
## [35] {turin}      => {foode}      0.02597403 1.0000000  0.02597403 38.5 12
## [36] {foode}      => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
## [37] {turin}      => {exchange}   0.02597403 1.0000000  0.02597403 38.5 12
## [38] {exchange}   => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
## [39] {turin}      => {aims}       0.02597403 1.0000000  0.02597403 38.5 12
## [40] {aims}       => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
## [41] {turin}      => {diets}      0.02597403 1.0000000  0.02597403 38.5 12
## [42] {diets}      => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
## [43] {turin}      => {dedicated}  0.02597403 1.0000000  0.02597403 38.5 12
## [44] {dedicated}  => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
## [45] {turin}      => {created}    0.02597403 1.0000000  0.02597403 38.5 12
## [46] {created}    => {turin}      0.02597403 1.0000000  0.02597403 38.5 12
```

```
## [47] {turin}      => {cohesion}   0.02597403 1.0000000  0.02597403 38.5 12

## [48] {cohesion}   => {turin}      0.02597403 1.0000000  0.02597403 38.5 12

## [49] {social}     => {practices}  0.02597403 1.0000000  0.02597403 38.5 12

## [50] {practices}  => {social}     0.02597403 1.0000000  0.02597403 38.5 12
```

Run code when error occurs

```
detach("package:arulesViz", unload=TRUE)
detach("package:arules", unload=TRUE)
library(arules)

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode

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

library(arulesViz) ## After arules works
library(igraph)
```

Using NetworkD3 To View Results

Build node and egdes properly formatted data files

Build the edgeList which will have SourceName, TargetName, Weight, SourceID, and TargetID

Convert the RULES to a DATAFRAME

```
Rules_DF2<-DATAFRAME(TweetTrans_rules, separate = TRUE)
(head(Rules_DF2))

##            LHS          RHS    support confidence   coverage lift count
## 2   {eatright}    {doesn't} 0.02380952          1 0.02380952   42    11
## 3    {doesn't}   {eatright} 0.02380952          1 0.02380952   42    11
## 4   {eatright} {frwpphvuub} 0.02380952          1 0.02380952   42    11
## 5 {frwpphvuub}   {eatright} 0.02380952          1 0.02380952   42    11
## 6   {eatright}   {straight} 0.02380952          1 0.02380952   42    11
## 7   {straight}   {eatright} 0.02380952          1 0.02380952   42    11

str(Rules_DF2)
```

```
## 'data.frame':    50 obs. of  7 variables:
##  $ LHS       : Factor w/ 16 levels "{eatright}","{doesn't}",..: 1 2 1 3 1
4 1 5 2 3 ...
##  $ RHS       : Factor w/ 16 levels "{doesn't}","{eatright}",..: 1 2 3 2 4
2 5 2 3 1 ...
##  $ support   : num  0.0238 0.0238 0.0238 0.0238 0.0238 ...
##  $ confidence: num  1 1 1 1 1 1 1 1 1 1 ...
##  $ coverage  : num  0.0238 0.0238 0.0238 0.0238 0.0238 ...
##  $ lift      : num  42 42 42 42 42 42 42 42 42 42 ...
##  $ count     : int  11 11 11 11 11 11 11 11 11 11 ...
```

Convert to char

```
Rules_DF2$LHS<-as.character(Rules_DF2$LHS)
Rules_DF2$RHS<-as.character(Rules_DF2$RHS)
```

Remove all

```
Rules_DF2[] <- lapply(Rules_DF2, gsub, pattern='[{]', replacement='')
Rules_DF2[] <- lapply(Rules_DF2, gsub, pattern='[}]', replacement='')
```

Remove the sup, conf, and count

USING LIFT

```
Rules_L<-Rules_DF2[c(1,2,5)]
names(Rules_L) <- c("SourceName", "TargetName", "Weight")
head(Rules_L,30)
```

```
##    SourceName TargetName             Weight
## 2    eatright    doesn't 0.0238095238095238
## 3     doesn't   eatright 0.0238095238095238
## 4    eatright frwpphvuub 0.0238095238095238
## 5  frwpphvuub   eatright 0.0238095238095238
## 6    eatright   straight 0.0238095238095238
## 7    straight   eatright 0.0238095238095238
## 8    eatright    soulful 0.0238095238095238
## 9     soulful   eatright 0.0238095238095238
## 20    doesn't frwpphvuub 0.0238095238095238
## 21 frwpphvuub    doesn't 0.0238095238095238
## 22    doesn't   straight 0.0238095238095238
## 23   straight    doesn't 0.0238095238095238
## 24    doesn't    soulful 0.0238095238095238
## 25    soulful    doesn't 0.0238095238095238
## 36 frwpphvuub   straight 0.0238095238095238
## 37   straight frwpphvuub 0.0238095238095238
## 38 frwpphvuub    soulful 0.0238095238095238
## 39    soulful frwpphvuub 0.0238095238095238
## 50   straight    soulful 0.0238095238095238
## 51    soulful   straight 0.0238095238095238
## 10   eatright       keep 0.0238095238095238
## 11       keep   eatright  0.025974025974026
## 26    doesn't       keep 0.0238095238095238
```

```
## 27       keep    doesn’t  0.025974025974026
## 40 frwpphvuub       keep 0.0238095238095238
## 41       keep frwpphvuub  0.025974025974026
## 52   straight       keep 0.0238095238095238
## 53       keep   straight  0.025974025974026
## 62    soulful       keep 0.0238095238095238
## 63       keep    soulful  0.025974025974026
```

USING SUP

```
Rules_S<-Rules_DF2[c(1,2,3)]
names(Rules_S) <- c("SourceName", "TargetName", "Weight")
head(Rules_S,30)
```

```
##    SourceName TargetName             Weight
## 2    eatright    doesn’t 0.0238095238095238
## 3     doesn’t   eatright 0.0238095238095238
## 4    eatright frwpphvuub 0.0238095238095238
## 5  frwpphvuub   eatright 0.0238095238095238
## 6    eatright   straight 0.0238095238095238
## 7    straight   eatright 0.0238095238095238
## 8    eatright    soulful 0.0238095238095238
## 9     soulful   eatright 0.0238095238095238
## 20    doesn’t frwpphvuub 0.0238095238095238
## 21 frwpphvuub    doesn’t 0.0238095238095238
## 22    doesn’t   straight 0.0238095238095238
## 23   straight    doesn’t 0.0238095238095238
## 24    doesn’t    soulful 0.0238095238095238
## 25    soulful    doesn’t 0.0238095238095238
## 36 frwpphvuub   straight 0.0238095238095238
## 37   straight frwpphvuub 0.0238095238095238
## 38 frwpphvuub    soulful 0.0238095238095238
## 39    soulful frwpphvuub 0.0238095238095238
## 50   straight    soulful 0.0238095238095238
## 51    soulful   straight 0.0238095238095238
## 10   eatright       keep 0.0238095238095238
## 11       keep   eatright 0.0238095238095238
## 26    doesn’t       keep 0.0238095238095238
## 27       keep    doesn’t 0.0238095238095238
## 40 frwpphvuub       keep 0.0238095238095238
## 41       keep frwpphvuub 0.0238095238095238
## 52   straight       keep 0.0238095238095238
## 53       keep   straight 0.0238095238095238
## 62    soulful       keep 0.0238095238095238
## 63       keep    soulful 0.0238095238095238
```

USING CONF

```
Rules_C<-Rules_DF2[c(1,2,4)]
names(Rules_C) <- c("SourceName", "TargetName", "Weight")
head(Rules_C,30)
```

```
##    SourceName TargetName             Weight
## 2    eatright    doesn't                  1
## 3     doesn't   eatright                  1
## 4    eatright frwpphvuub                  1
## 5  frwpphvuub   eatright                  1
## 6    eatright   straight                  1
## 7    straight   eatright                  1
## 8    eatright    soulful                  1
## 9     soulful   eatright                  1
## 20    doesn't frwpphvuub                  1
## 21 frwpphvuub    doesn't                  1
## 22    doesn't   straight                  1
## 23   straight    doesn't                  1
## 24    doesn't    soulful                  1
## 25    soulful    doesn't                  1
## 36 frwpphvuub   straight                  1
## 37   straight frwpphvuub                  1
## 38 frwpphvuub    soulful                  1
## 39    soulful frwpphvuub                  1
## 50   straight    soulful                  1
## 51    soulful   straight                  1
## 10   eatright       keep                  1
## 11       keep   eatright 0.916666666666667
## 26    doesn't       keep                  1
## 27       keep    doesn't 0.916666666666667
## 40 frwpphvuub       keep                  1
## 41       keep frwpphvuub 0.916666666666667
## 52   straight       keep                  1
## 53       keep   straight 0.916666666666667
## 62    soulful       keep                  1
## 63       keep    soulful 0.916666666666667
```

CHoose and set

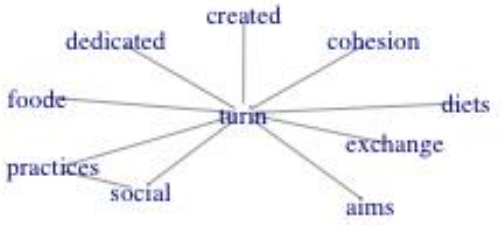
```
Rules_Sup<-Rules_C
Rules_Sup<-Rules_L
Rules_Sup<-Rules_S
```

Build a NetworkD3 edgeList and nodeList

```
edgeList<-Rules_Sup
```

Create a graph. Use simplyfy to ensure that there are no duplicated edges or self loops

```
MyGraph <- igraph::simplify(igraph::graph.data.frame(edgeList, directed=F))
plot(MyGraph, layout =  layout.auto, vertex.size = 9,
edge.arrow.size = 20, vertex.label.cex = 0.7, vertex.color = "white", vertex.
shape = "none")
```



BUILD THE NODES & EDGES

```
(edgeList<-Rules_Sup)
```

```
##     SourceName TargetName             Weight
## 2     eatright    doesn’t 0.0238095238095238
## 3      doesn’t   eatright 0.0238095238095238
## 4     eatright frwpphvuub 0.0238095238095238
## 5   frwpphvuub   eatright 0.0238095238095238
## 6     eatright   straight 0.0238095238095238
## 7     straight   eatright 0.0238095238095238
## 8     eatright    soulful 0.0238095238095238
## 9      soulful   eatright 0.0238095238095238
## 20     doesn’t frwpphvuub 0.0238095238095238
## 21  frwpphvuub    doesn’t 0.0238095238095238
## 22     doesn’t   straight 0.0238095238095238
## 23    straight    doesn’t 0.0238095238095238
## 24     doesn’t    soulful 0.0238095238095238
## 25     soulful    doesn’t 0.0238095238095238
## 36  frwpphvuub   straight 0.0238095238095238
## 37    straight frwpphvuub 0.0238095238095238
## 38  frwpphvuub    soulful 0.0238095238095238
## 39     soulful frwpphvuub 0.0238095238095238
## 50    straight    soulful 0.0238095238095238
```

```
## 51     soulful   straight 0.0238095238095238
## 10    eatright       keep 0.0238095238095238
## 11        keep   eatright 0.0238095238095238
## 26     doesn't       keep 0.0238095238095238
## 27        keep    doesn't 0.0238095238095238
## 40  frwpphvuub       keep 0.0238095238095238
## 41        keep frwpphvuub 0.0238095238095238
## 52    straight       keep 0.0238095238095238
## 53        keep   straight 0.0238095238095238
## 62     soulful       keep 0.0238095238095238
## 63        keep    soulful 0.0238095238095238
## 131      turin     social  0.025974025974026
## 132     social      turin  0.025974025974026
## 133      turin  practices  0.025974025974026
## 134  practices      turin  0.025974025974026
## 135      turin      foode  0.025974025974026
## 136      foode      turin  0.025974025974026
## 137      turin   exchange  0.025974025974026
## 138   exchange      turin  0.025974025974026
## 139      turin       aims  0.025974025974026
## 140       aims      turin  0.025974025974026
## 141      turin      diets  0.025974025974026
## 142      diets      turin  0.025974025974026
## 143      turin  dedicated  0.025974025974026
## 144  dedicated      turin  0.025974025974026
## 145      turin    created  0.025974025974026
## 146    created      turin  0.025974025974026
## 147      turin   cohesion  0.025974025974026
## 148   cohesion      turin  0.025974025974026
## 166     social  practices  0.025974025974026
## 167  practices     social  0.025974025974026
```

```
(MyGraph <- igraph::simplify(igraph::graph.data.frame(edgeList, directed=TRUE
)))
```

```
## IGRAPH ab719f5 DN-- 16 50 --
## + attr: name (v/c)
## + edges from ab719f5 (vertex names):
##  [1] eatright  ->doesn't    eatright  ->frwpphvuub eatright  ->straight
##  [4] eatright  ->soulful    eatright  ->keep       doesn't   ->eatright
##  [7] doesn't   ->frwpphvuub doesn't   ->straight   doesn't   ->soulful
## [10] doesn't   ->keep       frwpphvuub->eatright   frwpphvuub->doesn't
## [13] frwpphvuub->straight   frwpphvuub->soulful    frwpphvuub->keep
## [16] straight  ->eatright   straight  ->doesn't    straight  ->frwpphvuub
## [19] straight  ->soulful    straight  ->keep       soulful   ->eatright
## [22] soulful   ->doesn't    soulful   ->frwpphvuub soulful   ->straight
## + ... omitted several edges
```

```
nodeList <- data.frame(ID = c(0:(igraph::vcount(MyGraph) - 1)),
                       #networkD3 library requires IDs to start at 0
                       nName = igraph::V(MyGraph)$name)
```

Node Degree

```
(nodeList <- cbind(nodeList, nodeDegree=igraph::degree(MyGraph,
                                                       v = igraph::V(MyGraph)
, mode = "all")))
```

```
##            ID      nName nodeDegree
## eatright    0   eatright         10
## doesn't     1    doesn't         10
## frwpphvuub  2 frwpphvuub         10
## straight    3   straight         10
## soulful     4    soulful         10
## keep        5       keep         10
## turin       6      turin         18
## social      7     social          4
## practices   8  practices          4
## foode       9      foode          2
## exchange   10   exchange          2
## aims       11       aims          2
## diets      12      diets          2
## dedicated  13  dedicated          2
## created    14    created          2
## cohesion   15   cohesion          2
```

Betweenness

```
BetweenNess <- igraph::betweenness(MyGraph,
                                   v = igraph::V(MyGraph),
                                   directed = TRUE)

(nodeList <- cbind(nodeList, nodeBetweenness=BetweenNess))
```

```
##            ID      nName nodeDegree nodeBetweenness
## eatright    0   eatright         10               0
## doesn't     1    doesn't         10               0
## frwpphvuub  2 frwpphvuub         10               0
## straight    3   straight         10               0
## soulful     4    soulful         10               0
## keep        5       keep         10               0
## turin       6      turin         18              70
## social      7     social          4               0
## practices   8  practices          4               0
## foode       9      foode          2               0
## exchange   10   exchange          2               0
## aims       11       aims          2               0
## diets      12      diets          2               0
## dedicated  13  dedicated          2               0
```

```
## created    14    created          2               0
## cohesion   15   cohesion          2               0
```

BUILD THE EDGES

```
# edgeList<-Rules_Sup
getNodeID <- function(x){
  which(x == igraph::V(MyGraph)$name) - 1  #IDs start at 0
}
(getNodeID("ready"))

## numeric(0)

edgeList <- plyr::ddply(
  Rules_Sup, .variables = c("SourceName", "TargetName" , "Weight"),
  function (x) data.frame(SourceID = getNodeID(x$SourceName),
                          TargetID = getNodeID(x$TargetName)))

head(edgeList)

##   SourceName TargetName             Weight SourceID TargetID
## 1       aims      turin  0.025974025974026       11        6
## 2   cohesion      turin  0.025974025974026       15        6
## 3    created      turin  0.025974025974026       14        6
## 4  dedicated      turin  0.025974025974026       13        6
## 5      diets      turin  0.025974025974026       12        6
## 6    doesn't   eatright 0.0238095238095238        1        0

nrow(edgeList)

## [1] 50
```

Dice Sim

Calculate Dice similarities between all pairs of nodes

The Dice similarity coefficient of two vertices is twice the number of common neighbors divided by the sum of the degrees of the vertices. Method dice calculates the pairwise Dice similarities for some (or all) of the vertices.

```
DiceSim <- igraph::similarity.dice(MyGraph, vids = igraph::V(MyGraph), mode =
 "all")
head(DiceSim)

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
[,14]
## [1,]  1.0  0.8  0.8  0.8  0.8  0.8    0    0    0     0     0     0     0
    0
## [2,]  0.8  1.0  0.8  0.8  0.8  0.8    0    0    0     0     0     0     0
```

```
    0
## [3,]  0.8  0.8  1.0  0.8  0.8  0.8    0    0    0     0     0     0     0
    0
## [4,]  0.8  0.8  0.8  1.0  0.8  0.8    0    0    0     0     0     0     0
    0
## [5,]  0.8  0.8  0.8  0.8  1.0  0.8    0    0    0     0     0     0     0
    0
## [6,]  0.8  0.8  0.8  0.8  0.8  1.0    0    0    0     0     0     0     0
    0
##      [,15] [,16]
## [1,]     0     0
## [2,]     0     0
## [3,]     0     0
## [4,]     0     0
## [5,]     0     0
## [6,]     0     0
```

Create data frame that contains the Dice similarity between any two vertices

```
F1 <- function(x) {data.frame(diceSim = DiceSim[x$SourceID +1, x$TargetID + 1
])}
```

Place a new column in edgeList with the Dice Sim

```
head(edgeList)
```

```
##   SourceName TargetName             Weight SourceID TargetID
## 1       aims      turin  0.025974025974026       11        6
## 2   cohesion      turin  0.025974025974026       15        6
## 3    created      turin  0.025974025974026       14        6
## 4  dedicated      turin  0.025974025974026       13        6
## 5      diets      turin  0.025974025974026       12        6
## 6    doesn't   eatright 0.0238095238095238        1        0
```

```
edgeList <- plyr::ddply(edgeList,
                        .variables=c("SourceName", "TargetName", "Weight",
                                     "SourceID", "TargetID"),
                        function(x) data.frame(F1(x)))
head(edgeList)
```

```
##   SourceName TargetName             Weight SourceID TargetID diceSim
## 1       aims      turin  0.025974025974026       11        6     0.0
## 2   cohesion      turin  0.025974025974026       15        6     0.0
## 3    created      turin  0.025974025974026       14        6     0.0
## 4  dedicated      turin  0.025974025974026       13        6     0.0
## 5      diets      turin  0.025974025974026       12        6     0.0
## 6    doesn't   eatright 0.0238095238095238        1        0     0.8
```

```
D3_network_Tweets <- networkD3::forceNetwork(
  Links = edgeList, # data frame that contains info about edges
  Nodes = nodeList, # data frame that contains info about nodes
  Source = "SourceID", # ID of source node
```

```
  Target = "TargetID", # ID of target node
  Value = "Weight", # value from the edge list (data frame) that will be used
 to value/weight relationship amongst nodes
  NodeID = "nName", # value from the node list (data frame) that contains nod
e description we want to use (e.g., node name)
  Nodesize = "nodeBetweenness",  # value from the node list (data frame) that
 contains value we want to use for a node size
  Group = "nodeDegree",  # value from the node list (data frame) that contain
s value we want to use for node color
  height = 900, # Size of the plot (vertical)
  width = 1200,  # Size of the plot (horizontal)
  fontSize = 20, # Font size
  linkDistance = networkD3::JS("function(d) { return d.value*300; }"), # Func
tion to determine distance between any two nodes, uses variables already defi
ned in forceNetwork function (not variables from a data frame)
  linkWidth = networkD3::JS("function(d) { return d.value*0.2; }"),# Function
 to determine link/edge thickness, uses variables already defined in forceNet
work function (not variables from a data frame)
  opacity = 5, # opacity
  zoom = TRUE, # ability to zoom when click on the node
  opacityNoHover = 5, # opacity of labels when static
  linkColour = "brown")
```

Plot network

D3_network_Tweets

Save network as html file