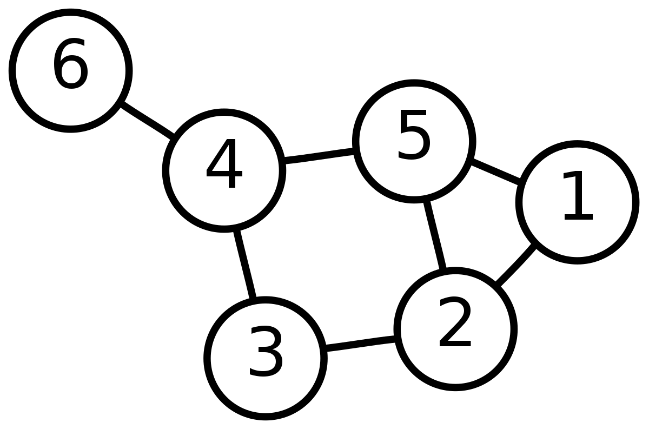
Q1. calculates Jaccard similarities and Cosine similarities for all pairs of vertices in the graph below. (you can do it manually or use R code, procedures or source code needed)



Answer:

Vector

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 2 | 1 | 0 | 1 | 0 | 1 | 0 |
| 3 | 0 | 1 | 0 | 1 | 0 | 0 |
| 4 | 0 | 0 | 1 | 0 | 1 | 1 |
| 5 | 1 | 1 | 0 | 1 | 0 | 0 |
| 6 | 0 | 0 | 0 | 1 | 0 | 0 |

1. Cosine

Manually

Example : cosine sim(1,2) = 1/sqrt(2)\*sqrt(3) = 0.48

R code

//code

install.packages("lsa")

library("lsa")

p1 = c( 0, 1, 0, 0, 1, 0 )

p2 = c( 1, 0, 1, 0, 1, 0 )

p3 = c( 0, 1, 0, 1, 0, 0 )

p4 = c( 0, 0, 1, 0, 1, 1 )

p5 = c( 1, 1, 0, 1, 0, 0 )

p6 = c( 0, 0, 0, 1, 0, 0 )

matrix = cbind(p1,p2, p3,p4,p5,p6)

cosine(matrix)

//result

p1 p2 p3 p4 p5 p6

p1 1.0000000 0.4082483 0.5000000 0.4082483 0.4082483 0.0000000

p2 0.4082483 1.0000000 0.0000000 0.6666667 0.3333333 0.0000000

p3 0.5000000 0.0000000 1.0000000 0.0000000 0.8164966 0.7071068

p4 0.4082483 0.6666667 0.0000000 1.0000000 0.0000000 0.0000000

p5 0.4082483 0.3333333 0.8164966 0.0000000 1.0000000 0.5773503

p6 0.0000000 0.0000000 0.7071068 0.0000000 0.5773503 1.0000000

1. Jaccard

Manually

Example : jaccard sim(1,2) = ¼ =0.25

R code

//code

Jaccard = function (x, y) { M.11 = sum(x == 1 & y == 1) M.10 = sum(x == 1 & y == 0)

M.01 = sum(x == 0 & y == 1) return (M.11 / (M.11 + M.10 + M.01))

}

p1 = c( 0, 1, 0, 0, 1, 0 )

p2 = c( 1, 0, 1, 0, 1, 0 )

p3 = c( 0, 1, 0, 1, 0, 0 )

p4 = c( 0, 0, 1, 0, 1, 1 )

p5 = c( 1, 1, 0, 1, 0, 0 )

p6 = c( 0, 0, 0, 1, 0, 0 )

input.variables = data.frame(p1,p2,p3,p4,p5,p6)

m = matrix(data = NA, nrow = length(input.variables), ncol = length(input.variables))

for (r in 1:length(input.variables)) {

for (c in 1:length(input.variables)) {

if (c == r) {

m[r,c] = 1

} else if (c > r) {

m[r,c] = Jaccard(input.variables[,r], input.variables[,c])

}

}

}

variable.names = sapply(input.variables, attr, "label")

colnames(m) = variable.names

rownames(m) = variable.names

jaccards = m

Result

| **NULL** | | **NULL** | | **NULL** | | **NULL** | | **NULL** | | **NULL** | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | |  | |  | |  | |  | |  |
| **NULL.** | 1 | | 0.25 | | 0.3333333 | | 0.25 | | 0.2500000 | | 0.0000000 |
| **NULL..1** | *NA* | | 1.00 | | 0.0000000 | | 0.50 | | 0.2000000 | | 0.0000000 |
| **NULL..2** | *NA* | | *NA* | | 1.0000000 | | 0.00 | | 0.6666667 | | 0.5000000 |
| **NULL..3** | *NA* | | *NA* | | *NA* | | 1.00 | | 0.0000000 | | 0.0000000 |
| **NULL..4** | *NA* | | *NA* | | *NA* | | *NA* | | 1.0000000 | | 0.3333333 |
| **NULL..5** | *NA* | | *NA* | | *NA* | | *NA* | | *NA* | | 1.000000 |

Q2. Extract 30 most recent tweets from twitter account of California Lutheran University (https://twitter.com/CalLutheran). Based on these tweets:

1. create a wordcloud (source code and wordcloud needed)

Answer:

install.packages("wordcloud",dependencies = TRUE)

library(wordcloud)

library(twitteR)

library(httpuv)

consumer\_key <- '4uIzbufjdqjxqc2pA2KTssKXZ'

consumer\_secret <- 'O40Ec4NYi3vb0kd5xQw2CxFeAfPsjnVm2jf4lH5uc2xQlPV9PI'

access\_token <- '1014453924667932672-TnssrO7NSr07LcUqC0ZRQY80XspeM4'

access\_secret <- 'cs6zX02SOaWMsD6fpEMEvUhpPpAKF8McYCgJ3w1enwa7u'

setup\_twitter\_oauth(consumer\_key, consumer\_secret, access\_token, access\_secret)

twitterUser <- getUser("CalLutheran")

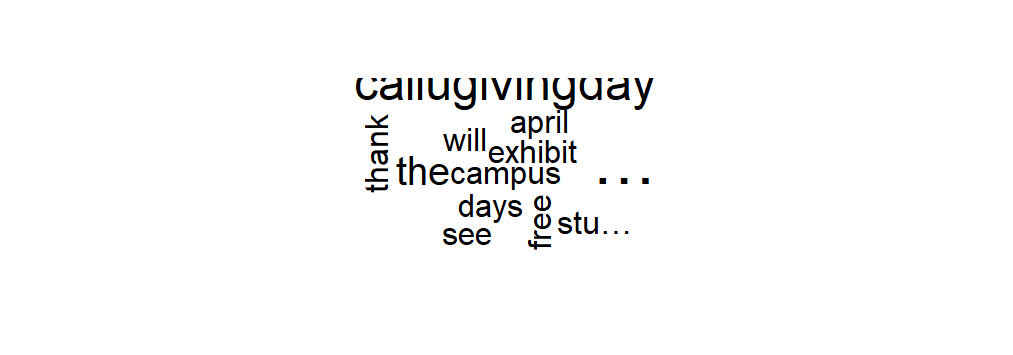
//I don’t know why I can’t 30 data even I adjust n=1000,I still only can get 26

tweets <- userTimeline(twitterUser, n = 1000)

my\_tweets\_text = sapply(tweets, function(x) x$getText())

wordcloud(my\_tweets\_text)

Word cloud



2. perform a sentiment analysis using “syuzhet” package, plot scores on a histogram. x-Axis displays sentiment score, y-Axis displays number of posts (source code and histogram needed)

Answer:

//use twitter, httpuv, tm, syuzhet, ggplot2

consumer\_key <- '4uIzbufjdqjxqc2pA2KTssKXZ'

consumer\_secret <- 'O40Ec4NYi3vb0kd5xQw2CxFeAfPsjnVm2jf4lH5uc2xQlPV9PI'

access\_token <- '1014453924667932672-TnssrO7NSr07LcUqC0ZRQY80XspeM4'

access\_secret <- 'cs6zX02SOaWMsD6fpEMEvUhpPpAKF8McYCgJ3w1enwa7u'

setup\_twitter\_oauth(consumer\_key, consumer\_secret, access\_token, access\_secret)

twitterUser <- getUser("CalLutheran")

tweets <- userTimeline(twitterUser, n = 1000)

df = do.call("rbind", lapply(tweets, as.data.frame))

encodeSentiment <- function(x) {

if(x <= -0.5){

"very negative"

}else if(x > -0.5 & x < 0){

"negative"

}else if(x > 0 & x < 0.5){

"positive"

}else if(x >= 0.5){

"very positive"

}else {

"neutral"

}

}

tweetSentiments <- get\_sentiment (df$text,method = "syuzhet")

df <- cbind(df, tweetSentiments)

df$sentiment <- sapply(tweets$tweetSentiments,encodeSentiment)

qplot(df$tweetSentiments) +theme(legend.position="none")+xlab("Sentiment Score") +ylab("Number of posts") +ggtitle("Tweets by Sentiment Score")

Sentimental analysis

