<http://www.evoketechnologies.com/blog/sentiment-analysis-r-language/>

<http://www.evoketechnologies.com/blog/sentiment-analysis-r-language/>

<https://www.r-bloggers.com/sentiment-analysis-on-donald-trump-using-r-and-tableau/>

SEntiAnalysis - Working Model

library(xlsl)

tweets = searchTwitter("#apple", n=200)

**tweets = searchTwitter("#iphone", n=1000, lang="en")**

Tweets.text = laply(tweets,function(t)t$getText())

pos = scan('c:/Users/gpran/Desktop/positive-words.txt', what='character', comment.char=';')

neg = scan('/Users/gpran/Desktop/negative-words.txt', what='character', comment.char=';')

score.sentiment = function(sentences, pos.words, neg.words, .progress='none')

{

require(plyr)

require(stringr)

# we got a vector of sentences. plyr will handle a list

# or a vector as an "l" for us

# we want a simple array ("a") of scores back, so we use

# "l" + "a" + "ply" = "laply":

scores = laply(sentences, function(sentence, pos.words, neg.words) {

# clean up sentences with R's regex-driven global substitute, gsub():

sentence = gsub('[[:punct:]]', '', sentence)

sentence = gsub('[[:cntrl:]]', '', sentence)

sentence = gsub('\\d+', '', sentence)

# and convert to lower case:

sentence = tolower(sentence)

# split into words. str\_split is in the stringr package

word.list = str\_split(sentence, '\\s+')

# sometimes a list() is one level of hierarchy too much

words = unlist(word.list)

# compare our words to the dictionaries of positive & negative terms

pos.matches = match(words, pos.words)

neg.matches = match(words, neg.words)

# match() returns the position of the matched term or NA

# we just want a TRUE/FALSE:

pos.matches = !is.na(pos.matches)

neg.matches = !is.na(neg.matches)

# and conveniently enough, TRUE/FALSE will be treated as 1/0 by sum():

score = sum(pos.matches) - sum(neg.matches)

return(score)

}, pos.words, neg.words, .progress=.progress )

scores.df = data.frame(score=scores, text=sentences)

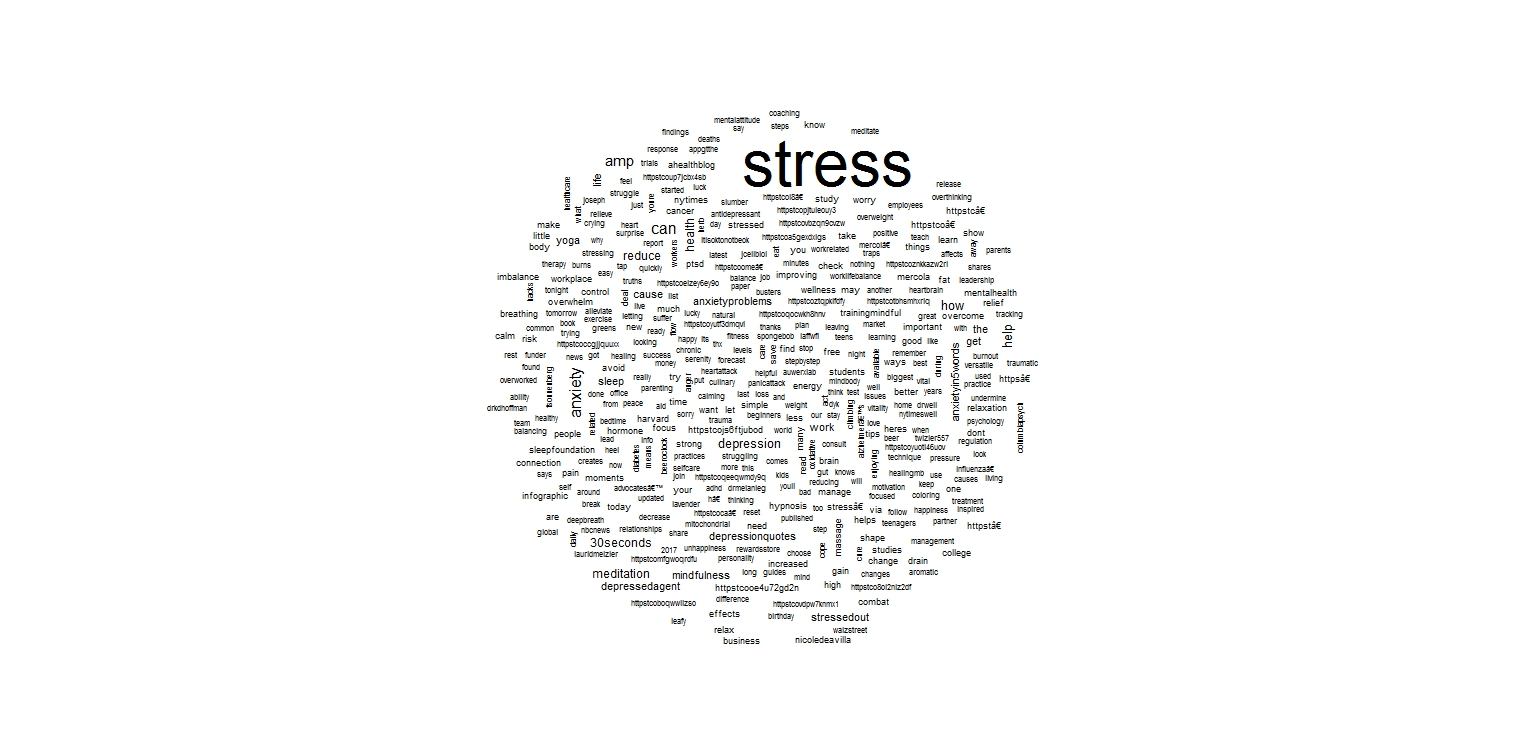
return(scores.df)

}

analysis = score.sentiment(Tweets.text, pos, neg)

table(analysis$score)

hist(analysis$score)



Ggmaps:

ggmaps:

v=VectorSource(stress$text)

docs=Corpus(v)

docs=tm\_map(docs, PlainTextDocument)

docs=tm\_map(docs, content\_transformer(function(x) iconv(x, to='ASCII', sub='byte')))

docs=tm\_map(docs, content\_transformer(function(x) tolower(x)))

docs <- tm\_map(docs, removeWords, stopwords('en'))

stripURL = function(x) {

gsub("www[^[:space:]]+|htt[^[:space:]]+", "", x)

}

docs <- tm\_map(docs, content\_transformer(stripURL))

docs <- tm\_map(docs, removeNumbers)

docs <- tm\_map(docs, removePunctuation)

docs <- tm\_map(docs, stripWhitespace)

stress$tweet <- as.character(unlist(sapply(docs, `[`, "content")))

stress$text <- NULL

allTweets <- stress %>%

group\_by(topic) %>%

distinct(., tweet)