rtweet package demo

CSCI0100 Fall 2022

Overview

- Need to create a twitter account (if you don’t already have one)
  - Verify your phone number
- https://apps.twitter.com/
  - Set up an app
  - Generate api keys and access tokens and keep them in a safe place

Install and include the proper packages

```
library(rtweet)
library(ggplot2)
library(dplyr)
library(tidytext)
```

Include your api keys and access token following this format (and then run it!)

```
# whatever name you assigned to your created app
appname <- "your-app-name"

## api key
key <- "yourLongApiKeyHere"

## api secret
secret <- "yourSecretKeyHere"

# create token named "twitter_token"
twitter_token <- create_token(
  app = your_twitter_username, # Your username
  consumer_key = "your api key here", # API key
  consumer_secret = "your secret api key here", # API Key Secret
  access_token = "your access token here", # Access Token
  access_secret = "your secret access token here" # Access Token Secret
)

auth_setup_default()
```

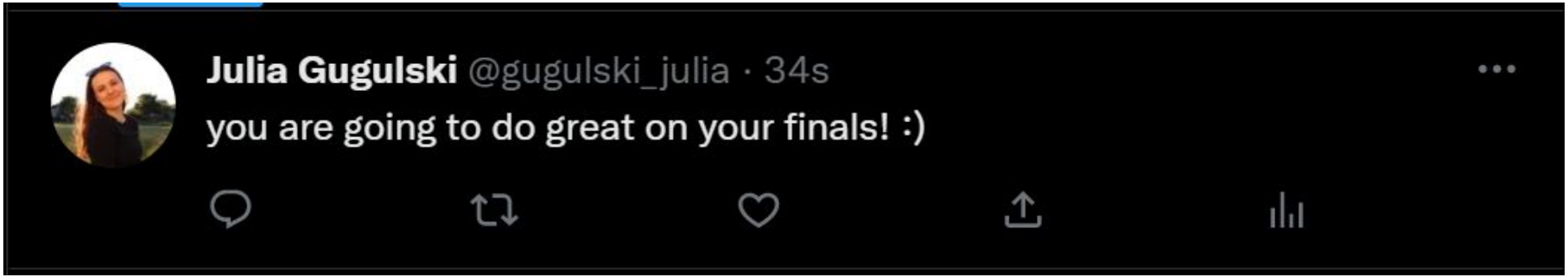
You should see these 4 variable populate in the environment

Cool functionality of rtweet... posting tweets from R!

```
# post a tweet from R
post_tweet("you are going to do great on your finals! :)")
## your tweet has been posted!
```

You should see this in your console: `Your tweet has been posted!`

And this was posted to my twitter!



Search for tweets on a topic you're interested in

I chose Obi-Wan to see what people on twitter thought about the new Obi-Wan show on Disney+

```
## search for Obi Wan related tweets, or choose any topic you're interested in
star_wars <- search_tweets(q = "Obi Wan",
                           n = 5000,
                           include_rts = FALSE) # removes retweets

# view the first 3 rows of the dataframe
head(star_wars, n = 3)

star_wars_tweets <- star_wars$full_text # take just the column of the
                                        # data frame that has text for text analysis
```

Sentiment Analysis

Text files containing positive and negative words

```
# Read in dictionaries of positive and negative words
pos <- scan('CS0100 TA/positive-words.txt',
            what = 'character',
            comment.char = ';') # change "CS0100 TA" to the
                                # name of the folder you saved txt file to
neg <- scan('CS0100 TA/negative-words.txt',
            what = 'character',
            comment.char = ';') # change "CS0100 TA" to the
                                # name of the folder you saved txt file to

# Add additional positive and negative words
pos <- c(pos, 'perf', 'luv', 'yum', 'epic', 'yay')
neg <- c(neg, 'wtf', 'ew', 'yuck')
```

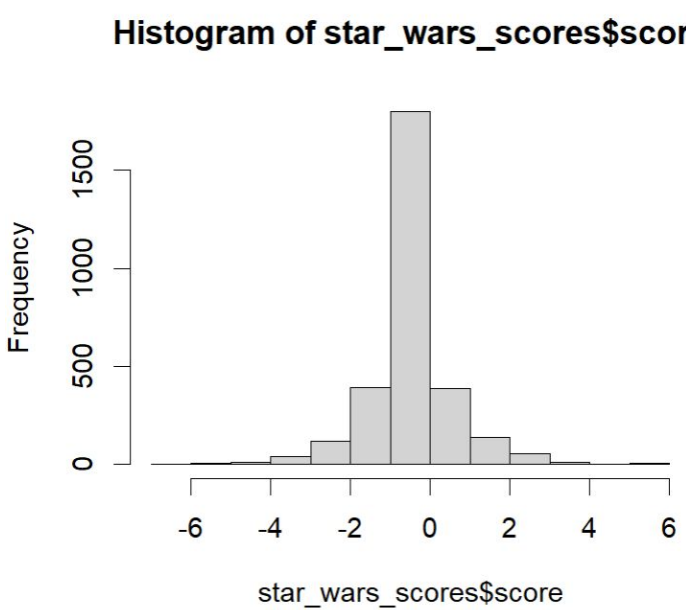
Function for scoring sentiment

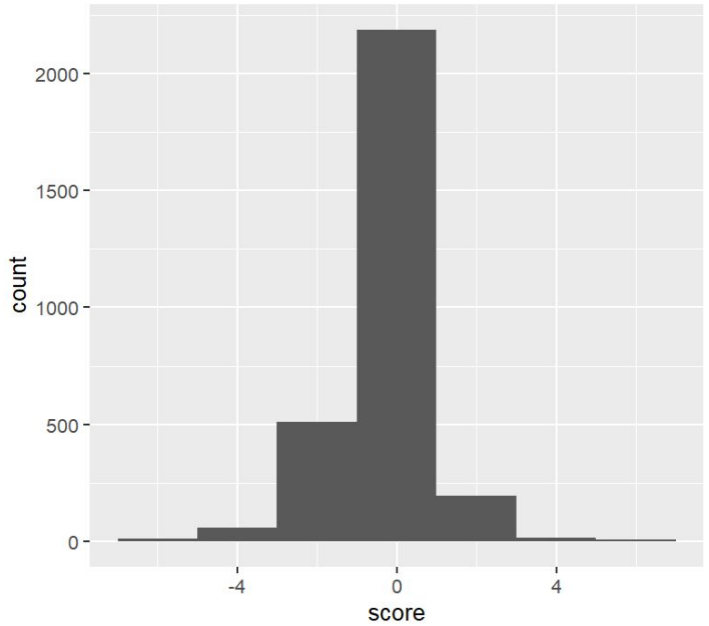
- Cleans the tweets that you collected using search_tweets()
- Matches positive/negative words
- Scores based on how many positive and negative words there are in a tweet
- Returns a data frame of the sentiment scores of all of the tweets you saved.

```r
# Define function for scoring sentiment
#source: https://jeffreybreen.wordpress.com/2011/07/04/twitter-text-mining-r-slides/
score_sentiment <- function(sentences, good_text, bad_text, .progress = 'none')
{
  require(plyr)
  require(stringr)
  scores <- laply(sentences, function(sentence, good_text, bad_text) {
    sentence <- gsub('[[:punct:]]', '', sentence) #remove punctuation
    sentence <- gsub('[[:cntrl:]]', '', sentence) #remove control characters
    sentence <- gsub('\\d+', '', sentence) #remove digits
    sentence <- iconv(sentence, 'UTF-8', 'ASCII', sub = "") #remove emojis
    sentence <- tolower(sentence) #make lowercase

    words = unlist(str_split(sentence, ' ')) #split sentence into words

    # compare our words to the dictionaries of positive & negative terms
    pos.matches = match(words, good_text)
    neg.matches = match(words, bad_text)

    # 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)
  }, good_text, bad_text, .progress = .progress )

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

Histogram from the scores

```
star_wars_scores <- score_sentiment(star_wars_tweets, pos, neg)
hist(star_wars_scores$score)
ggplot(star_wars_scores, aes(x=score)) + geom_histogram(binwidth = 2)
```





Observe the mean

```
> mean(star_wars_scores$score)
[1] 0.0191211
```

Keep in mind, this number will change every time you gather a new set of tweets!