Twitter Troll Project

Definition of troll

In my definition, Twitter trolls includes two categories:

(1) If a person's tweets often contain swear words.

(2) If a person's tweets often spark many comments that contain swear words. These tweets may not contain curse words, but they are provocative in nature. The comments, which is a large pool, are highly likely to contain swear words.

(3) People interacting with many trolls.

Collecting data

Retrieve user list: most followed Twitter accounts

We cannot work with all Twitter users since this pool is too big. Instead, we work with the 100 most followed users http://twittercounter.com/pages/100. Grab all users in this list:

First load packages:

```
library('RCurl')
```

```
## Loading required package: bitops
```

```
library('twitteR')
library('readr')
library(rvest)
```

```
## Loading required package: xml2
```

```
##
## Attaching package: 'rvest'
```

```
## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:twitteR':
##
##     id, location
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
```

```
setwd('/Users/guanghaoqi/Documents/Adv_data_science/twitter_troll_project')
accounts = read_lines("raw_data/top_twitter_accounts.txt")
accounts = accounts[grepl('@',accounts)]
accounts = gsub('.*@','',accounts)
accounts = gsub('Featured user:','',accounts)
```

Retrieve user list: 50 science stars on Twitter

The list is from the link: http://www.sciencemag.org/news/2014/09/top-50-science-stars-twitter.

```
top50_science_url = "http://www.sciencemag.org/news/2014/09/top-50-science-stars-twitter"
htmlfile = read_html(top50_science_url)

xpath.name = '//*[contains(concat( " ", @class, " " ), concat( " ", "k-index", " " )) and (((count(prec
nds.name = html_nodes(htmlfile, xpath = xpath.name)
top50sci.name = html_text(nds.name)

xpath.id = '//*[contains(concat( " ", @class, " " ), concat( " ", "k-index", " " )) and (((count(preced
nds.id = html_nodes(htmlfile, xpath = xpath.id)
top50sci.id = html_text(nds.id)
top50sci.id = gsub('@', '', top50sci.id)
```

Downloading tweets

Tutorial: R Twitter mining: https://www.youtube.com/watch?v=lT4Kosc_ers. First, get API authorization:

```
source('twitter_api.R')
setup_twitter_oauth(consumer_key, consumer_secret, access_token, access_secret)
```

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

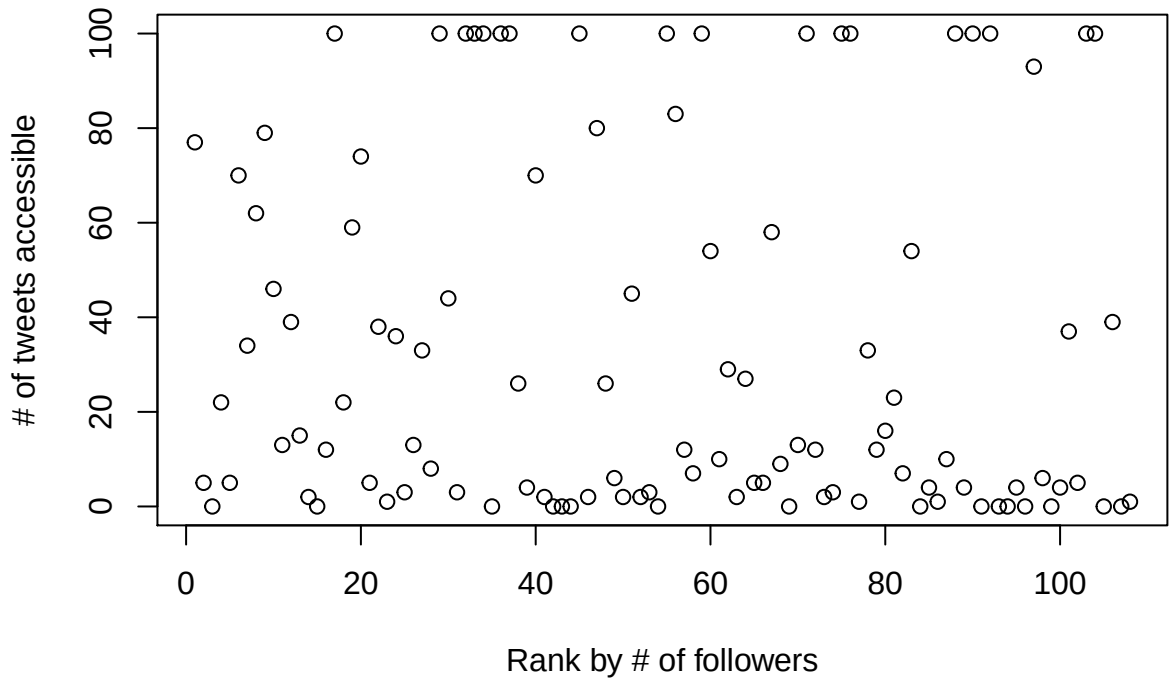
Next, search tweets based on the information above. For each account in the top 100 list, request 100 tweets from Twitter. This step was completed in multiple attempts due to API rate limits.

```
tweets = vector('list', length = length(accounts))
acct.search = paste0('from:',accounts)

# for (i in 1:length(accounts)){
#     tweets[[i]] = searchTwitter(acct.search[[i]], n = 5, lang = 'en', since = '2016-07-01', until = '
# }
# save(tweets, file = 'raw_data/tweets_top100.RData')
```

```
tweets.sci = vector('list', length = length(top50sci.id))
acct.search = paste0('from:', top50sci.id)

# for (i in 1:length(acct.search)){
#     tweets.sci[[i]] = searchTwitter(acct.search[i], n = 10, lang = 'en', since='2016-07-01', until='2
# }
# save(tweets.sci, file = 'raw_data/tweets_top50sci.RData')
```

Different users allow access to different numbers of tweets:

```
load('raw_data/tweets_top100.RData')
tweet_count = sapply(tweets,length)
plot(1:length(accounts),tweet_count, xlab = 'Rank by # of followers', ylab = '# of tweets accessible')
```



View texts

```
test = tweets[[1]][[1]]@.xData$text
print(test)
```

```
## [1] "Today, on #VoterRegistrationDay, I, @funnyordie and @rockthevote invite you to... #IRegistered
```

```
class(test)
```

```
## [1] "character"
```

English swear words list:

http://www.youswear.com/index.asp?language=English and http://www.noswearing.com/dictionary. I'm using the latter one because it's richer. Due to the small number of accessible tweets of some accounts, we may not identify any of the words in a small disctionary. We compute the frequency of swear words in each person's tweets.

```r
swear = read_lines('raw_data/swearwords.txt')
swear = swear[swear!='']
# Now clean the curse words dictionary swear and save it i vector swear.cl
swear.cl = swear %>% sapply(strsplit, split = ' - ') %>%
    unlist %>% unique
# head(swear.cl)
```

Frequency of the swear words: how to retrieve strings from S4 class?

Next step for data collection

Current problems in data collection

1. The tweets scraped with R are fewer than what I see on the browser.
2. Is there a rate limit if I scrape tweets without API?

Information about a particular tweet

Google that tweet, the first entry links to the page containing all the needed information, including the number of likes, the number of retweets and reply from other people.

Example: "Scientists use words differently than their usual meanings, causing confusion. Our future depends on a clear understanding of science.pic.twitter.com/Ol57Lba1X6"