


Twitter Mining with R

Dr. Shahid Mahmood Awan

github.com/shahidmawan

shahidmawan@gmail.com

Mining Twitter Data


 [Developers](#) [API Health](#) [Blog](#) [Discussions](#) [Documentation](#) [Sign in](#)

[Home](#)

Sign in with your Twitter account

Please log in to access that page.

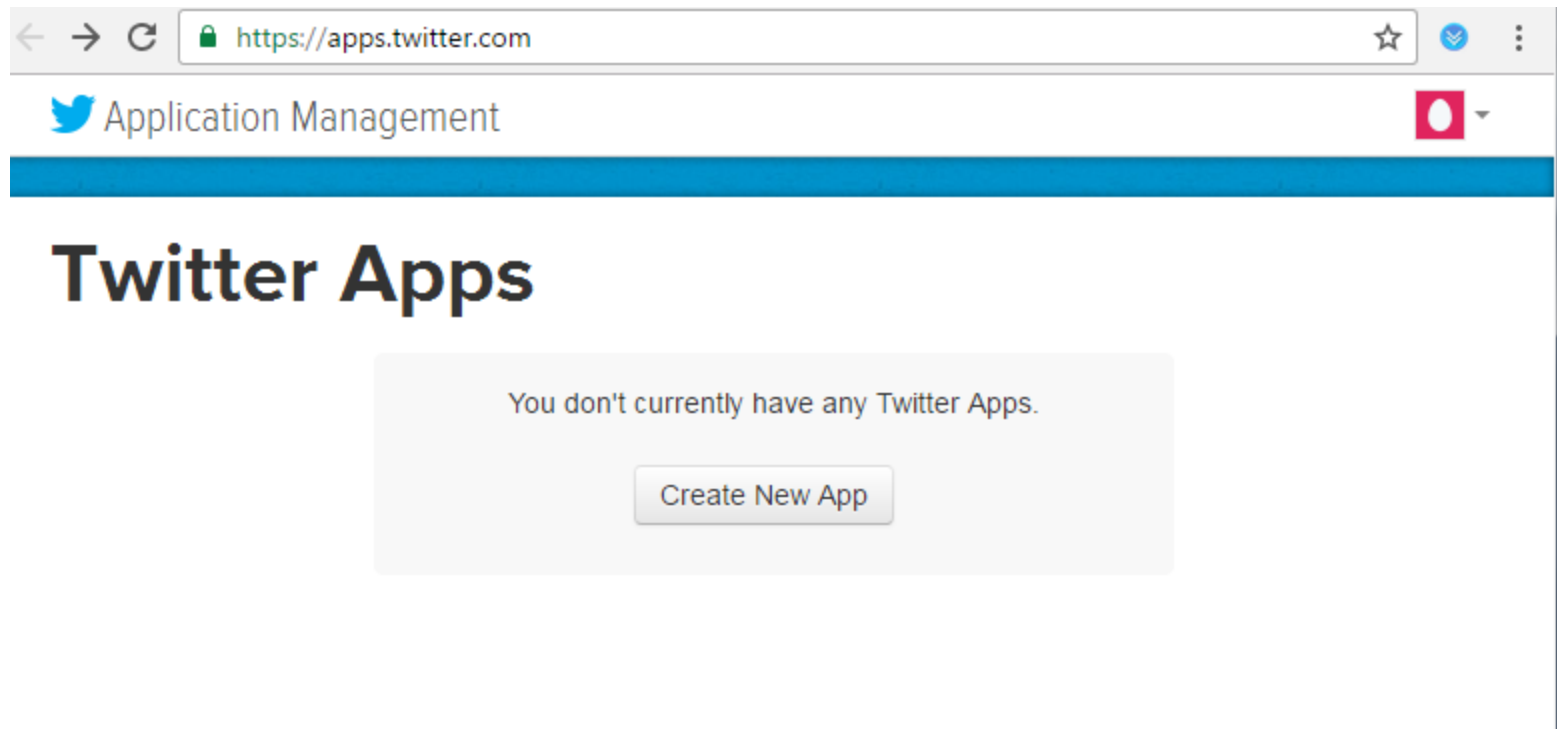
Username: *

New to Twitter? [Sign up](#) 

Password: *

[Log in](#)

Mining Twitter Data



Mining Twitter Data



 Application Management



Create an application

Application Details

Name *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This URL will be used for source attribution for tweets created by your application and will be shown in user-facing authorization screens.

Mining Twitter Data

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This URL is used for source attribution for tweets created by your application and will be shown in user-facing authorization screens. (If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

Where should we return after successfully authenticating? [OAuth 1.0a](#) applications should explicitly specify their `oauth_callback` URL here. To restrict your application from using callbacks, leave this field blank.

Developer Agreement

☐ Yes, I have read and agree to the [Twitter Developer Agreement](#).

Create your Twitter application

Mining Twitter Data

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This URL is used for source attribution for tweets created by your application and will be shown in user-facing authorization screens. (If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

Where should we return after successfully authenticating? [OAuth 1.0a](#) applications should explicitly specify their `oauth_callback` URL here. To restrict your application from using callbacks, leave this field blank.

Developer Agreement

☐ Yes, I have read and agree to the [Twitter Developer Agreement](#).

Create your Twitter application

Mining Twitter Data

← → ↻ <https://apps.twitter.com/app/13114542> ☆

 Application Management 

Your application has been created. Please take a moment to review and adjust your application's settings.

TAppTestDemo

Test OAuth

Details

Settings

Keys and Access Tokens

Permissions



Test Twitter APIs

<http://www.umt.edu.pk>

Organization

Information about the organization or company associated with your application. This information is optional.

Organization None

Organization website None

Mining Twitter Data

← → ↺ <https://apps.twitter.com/app/13114542/keys> ☆ ⌵

TAppTestDemo

Test OAuth

Details

Settings

Keys and Access Tokens

Permissions

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key) LQdcDF451pFHxwhcE60CwzqRn

Consumer Secret (API Secret) dWM7kSBJw0rjo2U1DS5pxB3oYjZm7XQRr120rbBlaH8W3DMUmn

Access Level Read and write ([modify app permissions](#))

Owner Shahid3.1415926535

Owner ID 7999526 ... 35159

Application Actions

Regenerate Consumer Key and Secret

Change App Permissions

Mining Twitter Data

https://apps.twitter.com/app/13117572/keys

Status
Your application's Consumer Key and Consumer Secret have been successfully regenerated.
[Refresh](#) if your changes are not yet indicated.

TAppTestDemo

Test OAuth

Details

Settings

Keys and Access Tokens

Permissions

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	mj2ybKiqzD6jaPwwF . . . af		
Consumer Secret (API Secret)	2iVYN1AG,	'TnaeaHHV7sA' "	.. WWdqfDRK13
Access Level	Read and write (modify app permissions)		
Owner	Shahid_ 365		
Owner ID	7999526- 5153		

Demo

- https://www.youtube.com/watch?v=IT4Kosc_ers

Links

- <https://sites.google.com/site/miningtwitter/basics>
- <https://sivaanalytics.wordpress.com/2013/10/10/sentiment-analysis-on-twitter-data-using-r-part-i/>
- <http://andybromberg.com/sentiment-analysis/>
- <http://www.dataperspective.info/2013/08/sentiment-analysis-using-r.html>
- <https://www.credera.com/blog/business-intelligence/twitter-analytics-using-r-part-1-extract-tweets/>
- <https://www.credera.com/blog/business-intelligence/twitter-analytics-using-r-part-2-create-word-cloud/>
- <https://www.credera.com/blog/technology-insights/open-source-technology-insights/twitter-analytics-using-r-part-3-compare-sentiments/>

Step 2: INSTALL AND LOAD R PACKAGES

- R comes with a standard set of packages. A number of other packages are [available for download](#) and installation. For the purpose of this post, we will need the following packages:
- – ROAuth: *Provides an interface to the OAuth 1.0 specification, allowing users to authenticate via OAuth to the server of their choice.*
- – Twitter: *Provides an interface to the Twitter web API.*
- Let's start by installing and loading all the required packages.
- `install.packages("twitteR")`
- `install.packages("ROAuth")`
- `library("twitteR")`
- `library("ROAuth")`

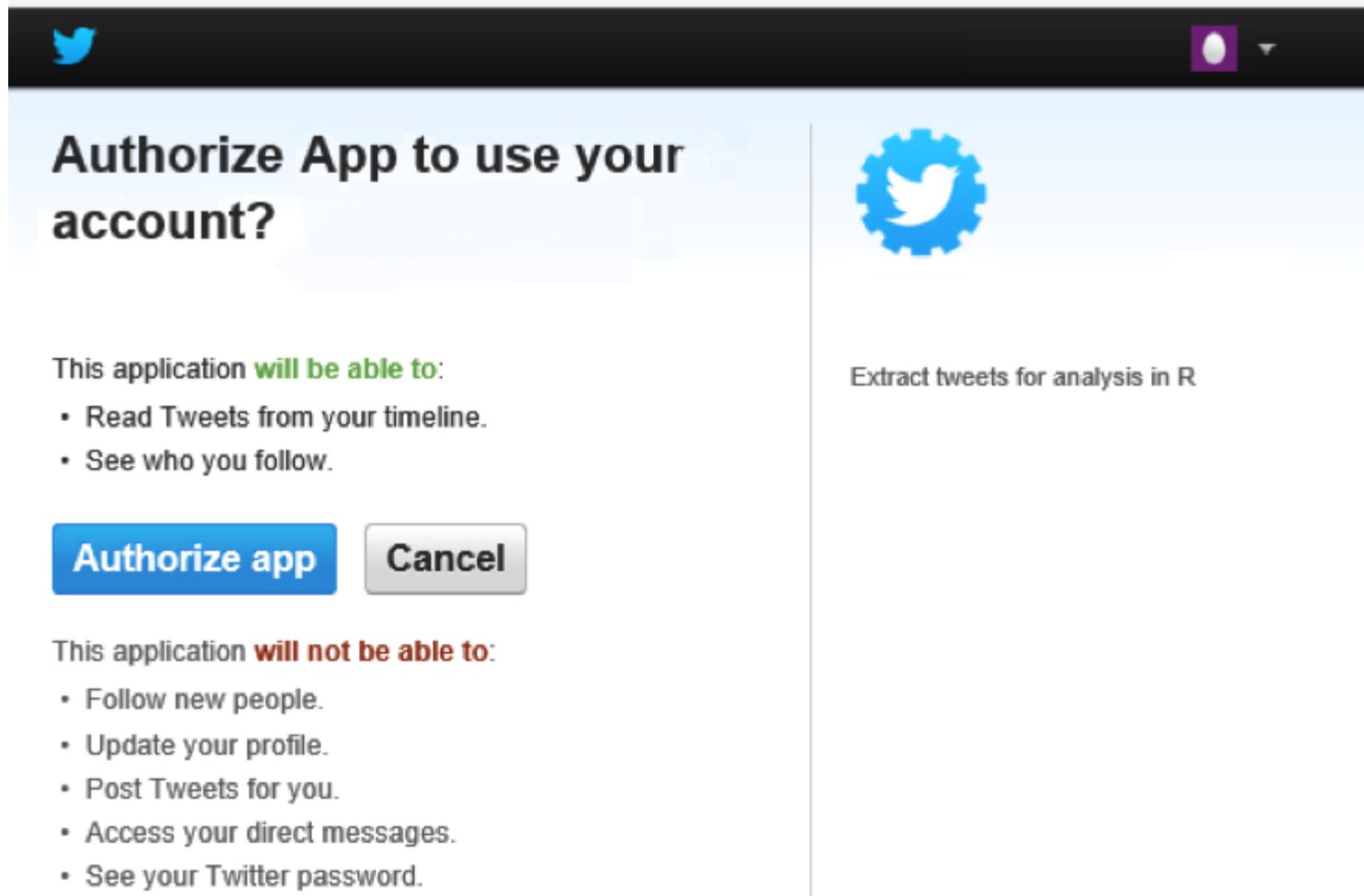
3. CREATE AND STORE TWITTER AUTHENTICATED CREDENTIAL OBJECT

- If you are a Windows user, you need to get “cacert.pem” file. Download the “cacert.pem” file from the specified URL and store it in your working directory. Then create an object “cred” that will save the authenticated object for later sessions and initiate the handshake. This is where you will enter the consumerKey and consumerSecret from the first step. Once the handshake is complete it will direct you to a hyperlink in the console window.

3. CREATE AND STORE TWITTER AUTHENTICATED CREDENTIAL OBJECT

- # Download "cacert.pem" file
download.file(url="http://curl.haxx.se/ca/cacert.pem",dest
file="cacert.pem") #create an object "cred" that will save
the authenticated object that we can use for later sessions
cred <-
OAuthFactory\$new(consumerKey='XXXXXXXXXXXXXXXXXXXXX',
consumerSecret='XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX',
requestURL='https://api.twitter.com/oauth/request_token',
accessURL='https://api.twitter.com/oauth/access_token',
authURL='https://api.twitter.com/oauth/authorize') #
Executing the next step generates an output --> To enable
the connection, please direct your web browser to:
<hyperlink> . Note: You only need to do this part once
cred\$handshake(cainfo="cacert.pem")

Navigate to the specified link to authorize app and click
“Authorize App”.





You've granted access to App!

Next, return to App and enter this PIN to complete the authorization process:

6383582

[← Go to Twitter](#)

[Go to the App homepage](#)

You can revoke access to any application at any time from the [Applications tab](#) of your Settings page.

By authorizing an application you continue to operate under [Twitter's Terms of Service](#). In particular, some usage information will be shared back with Twitter. For more, see our [Privacy Policy](#).

- #save for later use for Windows
- `save(cred, file="twitter authentication.Rdata")`

4. EXTRACT TWEETS

- Load “twitter authentication.Rdata” file in your session and run registerTwitterOAuth. This should return “TRUE” indicating that all is good and we can proceed. Then we set two variables, one for the search string, which could be a hashtag or user mention, and the second variable is the number of tweets we want to extract for analysis. Use searchTwitter to search Twitter based on the supplied search string and return a list. The “lang” parameter is used below to restrict tweets to the “English” language.

- `load("twitter authentication.Rdata")`
`registerTwitterOAuth(cred)` `search.string <-`
`"#nba"` `no.of.tweets <- 100` `tweets <-`
`searchTwitter(search.string, n=no.of.tweets,`
`cainfo="cacert.pem", lang="en")` `tweets`

Code

- `library(httr)`
- `library(devtools)`
- `library(twitteR)`
- `library(base64enc)`
- `library("ROAuth")`
- `options(httr_oauth_cache=T)`

- `consumer_key1<- 'mj2ybKiqzD6jaPwvFhXqmcGaf'`
- `consumer_secret1<-
'2iVYN1AQjG59BGKjOJm7TnaeaHHV7sAhO2vc2BDFWWdqfDRK13'`
- `access_token1<- '799952640897585153-HFeLThCF80NuDsWMDCu47TdJZBiegE5'`
- `access_token_secret<- 'iHfThoxuhaFzyAuiaHnrp34yCaehY73CibSqZEH4P4Rqh'`
- `setup_twitter_oauth(consumer_key1, consumer_secret1,
access_token=access_token1, access_secret=access_token_secret)`
- `#setup_twitter_oauth(consumer_key, consumer_secret, access_token=NULL,
access_secret=NULL)`
- `LFC_tweets <- searchTwitter("LFC", n=10, lang="en")`

Code

- `consumer_key1<- 'mj2ybKiqzD6jaPwvFhXqmcGaf'`
- `consumer_secret1<-
'2iVYN1AQjG59BGKjOJm7TnaeaHHV7sAhO2vc2BDFWWdqfDRK13'`
- `access_token1<- '799952640897585153-
HFeLThCF80NuDsWMDCu47TdJZBiegE5'`
- `access_token_secret<- 'iHfThoxuhaFzyAuiaHnrp34yCaehY73CibSqZEH4P4Rqh'`
- `setup_twitter_oauth(consumer_key1, consumer_secret1,
access_token=access_token1, access_secret=access_token_secret)`
- `#setup_twitter_oauth(consumer_key, consumer_secret, access_token=NULL,
access_secret=NULL)`
- `LFC_tweets <- searchTwitter("LFC", n=10, lang="en")`

Twitter Sentiment Analysis using R

- The implementation of the Review Engine will be as follows:
 - Gets Tweets from Twitter
 - Clean the data
 - Create a Word Cloud
 - Create a data dictionary
 - Score each tweet.

1. EXTRACT TWEETS

- Load the Twitter authentication and extract tweets using #nba.
- `load("twitter authentication.Rdata")`
`registerTwitterOAuth(cred)`
- `tweets <- searchTwitter("#nba", n=1499, cainfo="cacert.pem", lang="en")`
- `tweets.text <- sapply(tweets, function(x) x$getText())`

2. CLEAN UP TEXT

- We have already been authenticated and successfully retrieved the text from the tweets using #nba.
- The first step in creating a word cloud is to clean up the text by using lowercase and removing punctuation, usernames, links, etc.
- We are using the function gsub to replace unwanted text. Gsub will replace all occurrences of any given pattern.
- Although there are alternative packages that can perform this operation, we have chosen gsub because of its simplicity and readability.

2. CLEAN UP TEXT

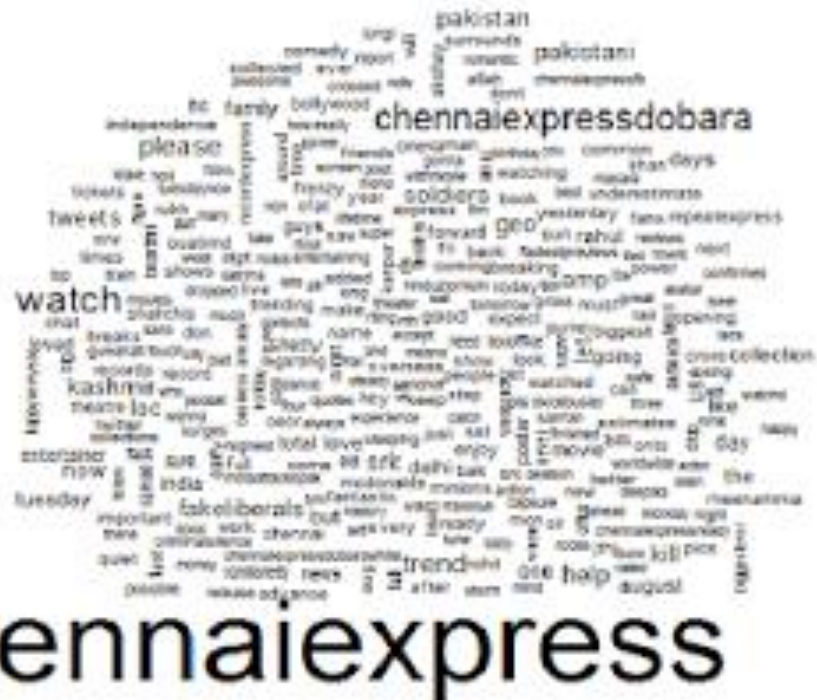
- #convert all text to lower case
- `tweets.text <- tolower(tweets.text)`
- # Replace blank space ("rt")
- `tweets.text <- gsub("rt", "", tweets.text)`
- # Replace @UserName
- `tweets.text <- gsub("@\\w+", "", tweets.text)`
- # Remove punctuation
- `tweets.text <- gsub("[[:punct:]]", "", tweets.text)`

2. CLEAN UP TEXT

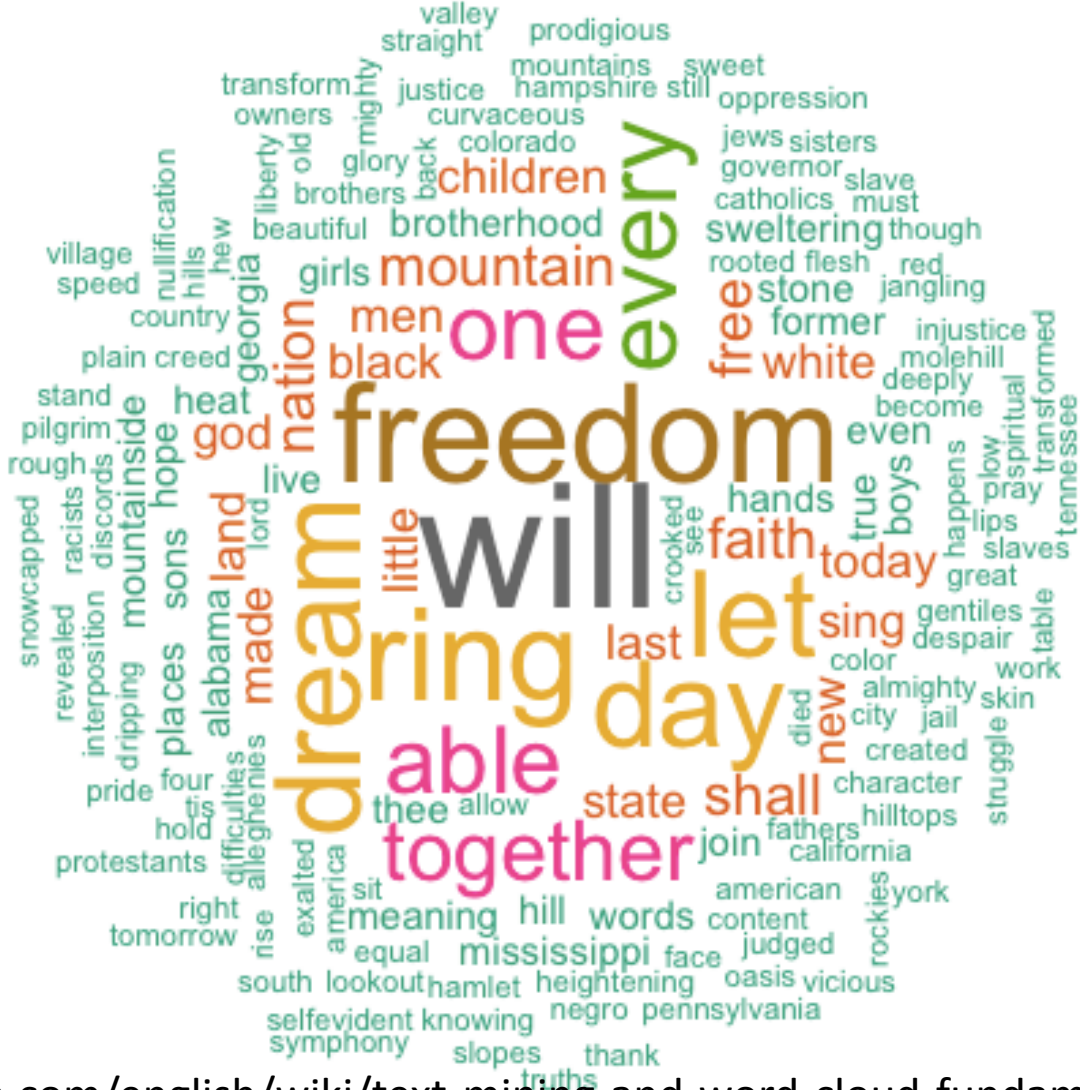
- # Remove links
- `tweets.text <- gsub("http\\w+", "", tweets.text)`
- # Remove tabs
- `tweets.text <- gsub("[|\\t]{2,}", "", tweets.text)`
- # Remove blank spaces at the beginning
- `tweets.text <- gsub("^ ", "", tweets.text)`
- # Remove blank spaces at the end
- `tweets.text <- gsub(" $", "", tweets.text)`

Create a Word Cloud:

- library(wordcloud)
- wordcloud(tweets_cl)
-



Word Cloud



<http://www.sthda.com/english/wiki/text-mining-and-word-cloud-fundamentals-in-r-5-simple-steps-you-should-know>

Install and load the required packages

- **text mining** and **wordcloud** packages are required.
- They can be installed and loaded using the R code below :
- # Install
- ```
install.packages("tm") # for text mining
install.packages("SnowballC") # for text stemming
install.packages("wordcloud") # word-cloud generator
install.packages("RColorBrewer") # color palettes
```
- # Load
- ```
library("tm")  
library("SnowballC")  
library("wordcloud")  
library("RColorBrewer")
```

Read the text file

- filePath <-
"http://www.sthda.com/sthda/RDoc/example-files/martin-luther-king-i-have-a-dream-speech.txt"
- text <- readLines(filePath)
- # Load the data as a corpus
- docs <- Corpus(VectorSource(text))

Text transformation

- Transformation is performed using **tm_map()** function to replace, for example, special characters from the text.
- Replacing “/”, “@” and “|” with space
- toSpace <- content_transformer(function (x , pattern) gsub(pattern, " ", x))
- docs <- tm_map(docs, toSpace, "/")
- docs <- tm_map(docs, toSpace, "@")
- docs <- tm_map(docs, toSpace, "\\|")

Cleaning the text

- the **tm_map()** function is used to remove unnecessary white space, to convert the text to lower case, to remove common stopwords like ‘the’, “we”.
- You could also remove numbers and punctuation with **removeNumbers** and **removePunctuation** arguments.

Text Cleaning

- The R code below can be used to clean your text :

Convert the text to lower case

- **docs <- tm_map(docs, content_transformer(tolower))**

Remove numbers

- **docs <- tm_map(docs, removeNumbers)**

Remove english common stopwords

- **docs <- tm_map(docs, removeWords, stopwords("english"))**

Remove your own stop word # specify your stopwords as a character vector

- **docs <- tm_map(docs, removeWords, c("blabla1", "blabla2"))**

Remove punctuations

- **docs <- tm_map(docs, removePunctuation)**

Eliminate extra white spaces

- **docs <- tm_map(docs, stripWhitespace)**

Text stemming

- **docs <- tm_map(docs, stemDocument)**

Build a term-document matrix

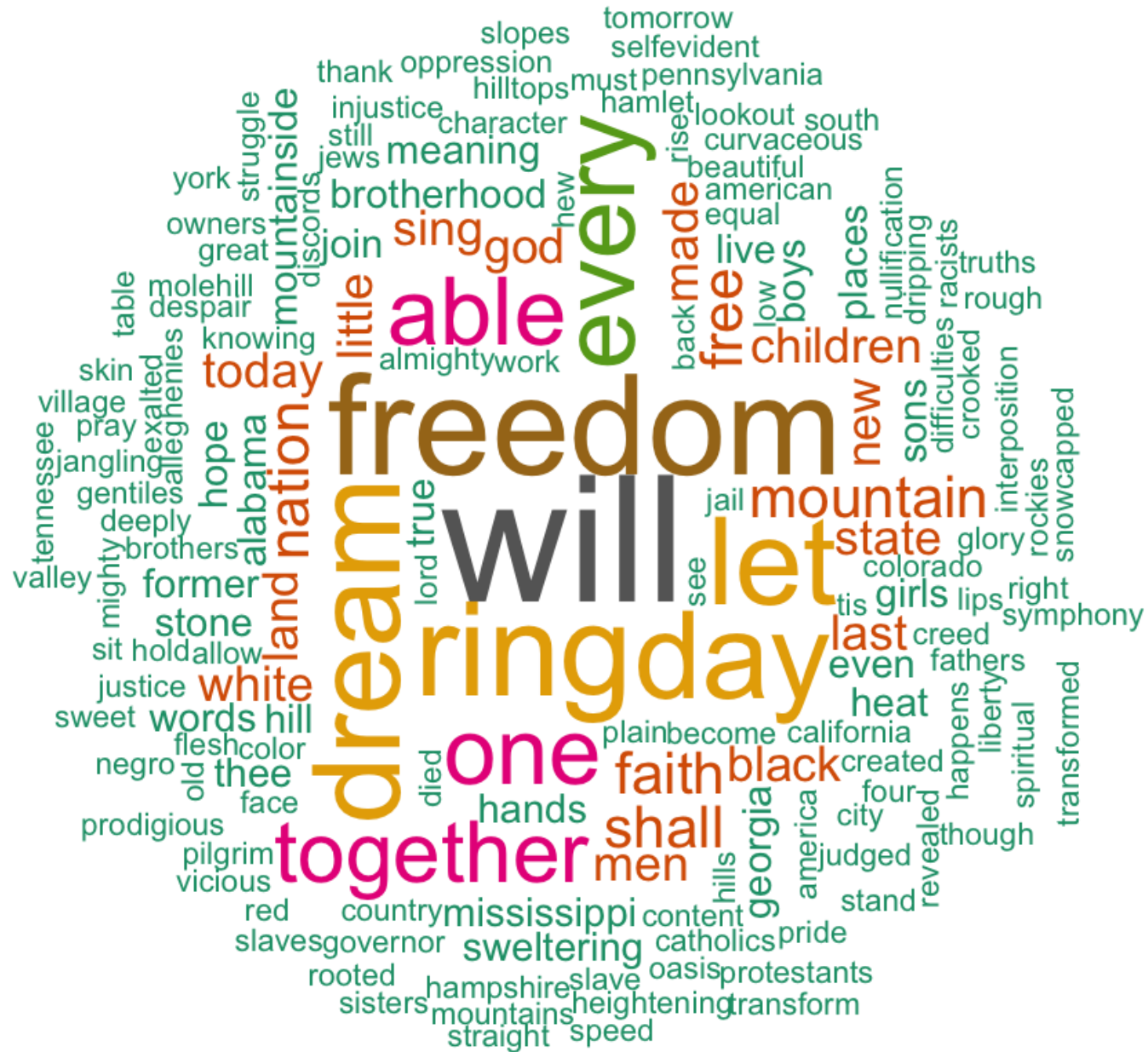
- `dtm <- TermDocumentMatrix(docs)`
- `m <- as.matrix(dtm)`
- `v <- sort(rowSums(m), decreasing=TRUE)`
- `d <- data.frame(word = names(v), freq=v)`
- **`head(d, 10)`**

Word Frequency

	word	freq
will	will	17
freedom	freedom	13
ring	ring	12
day	day	11
dream	dream	11
let	let	11
every	every	9
able	able	8
one	one	8
together	together	7

Generate the Word cloud

- `set.seed(1234)`
- `wordcloud(words = d$word, freq = d$freq,
min.freq = 1, max.words=200,
random.order=FALSE, rot.per=0.35,
colors=brewer.pal(8, "Dark2"))`



Explore frequent terms and their associations

- `findFreqTerms(dtm, lowfreq = 4)`
- `[1] "able" "day" "dream" "every" "faith" "free"`
`"freedom" "let" "mountain" "nation"`
- `[11] "one" "ring" "shall" "together" "will"`

- findAssocs(dtm, terms = "freedom", corlimit = 0.3)

Plot word frequencies

- The frequency of the first 10 frequent words are plotted :
- `barplot(d[1:10,]$freq, las = 2, names.arg = d[1:10,]$word, col = "lightblue", main = "Most frequent words", ylab = "Word frequencies")`

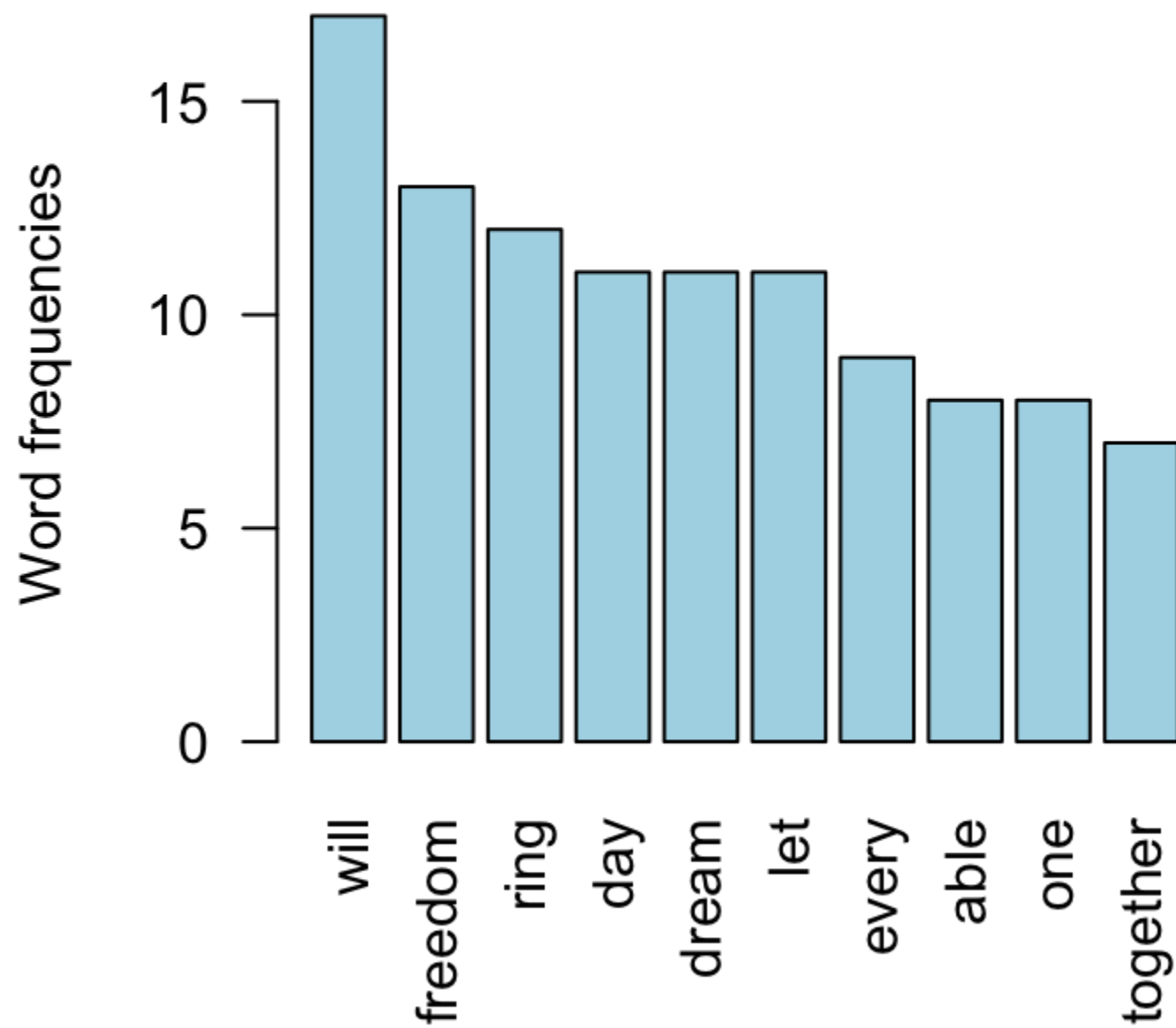
Create a data dictionary:

- In this step, we create use a Dictionary of words containing positive, negative words which are downloaded from [here](#). These 2 types of words are used as keywords for classifying the each tweet into one of the 4 categories: Very Positive, Positive, Negative and Very Negative.

Score each tweet:

- In this step, we will write a function which will calculate rating of the movie. The function is given below. After calculating the scores we plot graphs showing the rating as “WORST”, “BAD”, “GOOD”, “VERYGOOD”

Most frequent words

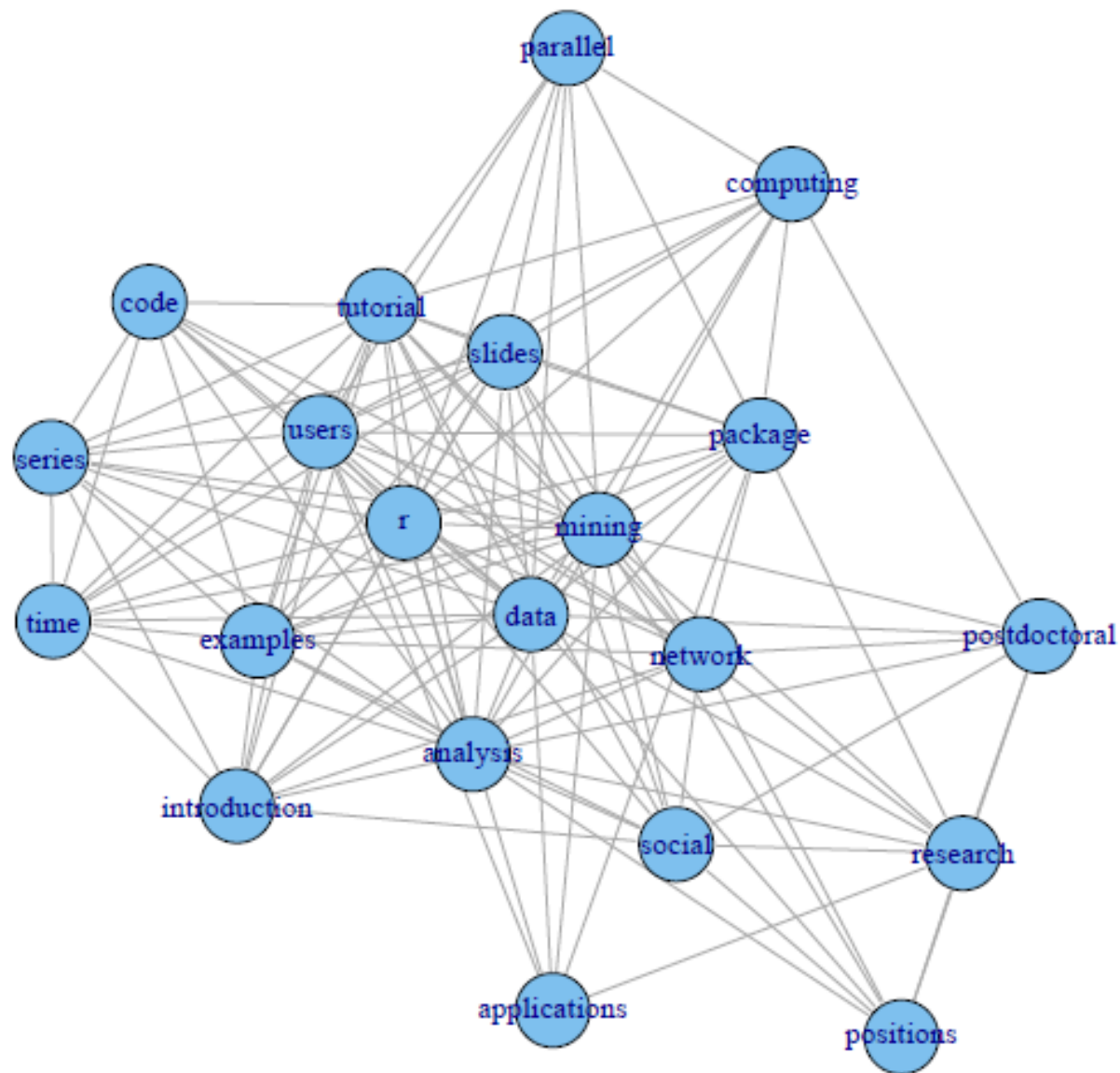


Build a Graph

- ```
> library(igraph)
> # build a graph from the above matrix
> g <- graph.adjacency(termMatrix,
weighted=T, mode = "undirected")
> # remove loops
> g <- simplify(g)
> # set labels and degrees of vertices
> V(g)$label <- V(g)$name
> V(g)$degree <- degree(g)
```

# Plot the Graph

- ```
> # set seed to make the layout reproducible  
> set.seed(3952)  
> layout1 <- layout.fruchterman.reingold(g)  
> plot(g, layout=layout1)
```



Sentiment Analysis

- # install package sentiment140
- require(devtools)
- install_github("okugami79/sentiment140", "okugami79")
- # sentiment analysis
- library(sentiment)
- sentiments <- sentiment(tweets.df\$text)
- table(sentiments\$polarity)
- ##
- ## neutral positive
- ## 428 20
- # sentiment plot
- sentiments\$score <- 0
- sentiments\$score[sentiments\$polarity == "positive"] <- 1
- sentiments\$score[sentiments\$polarity == "negative"] <- -1
- sentiments\$date <- as.IDate(tweets.df\$created)
- result <- aggregate(score ~ date, data = sentiments, sum)
- plot(result, type = "l")

links

- <http://www.sthda.com/english/wiki/text-mining-and-word-cloud-fundamentals-in-r-5-simple-steps-you-should-know>
- <http://stackoverflow.com/questions/20561120/creating-network-of-terms-of-maximum-occurring-terms-in-r-language>
- <http://www.rdatamining.com/examples/social-network-analysis>
- <http://www.rdatamining.com/examples/text-mining>

Applications

- <http://www.expertsystem.com/10-text-mining-examples/>

Research Project #1

Twitter Sentiment Analysis for Understanding Citizens' Trust in Government

- Collected over 1m tweets from January 2013 from 60 accounts
 - 20 cities, 20 mayors, 20 police departments
- Analysis was done using R (for data retrieval, preparation, and computation) and Excel (for plotting)

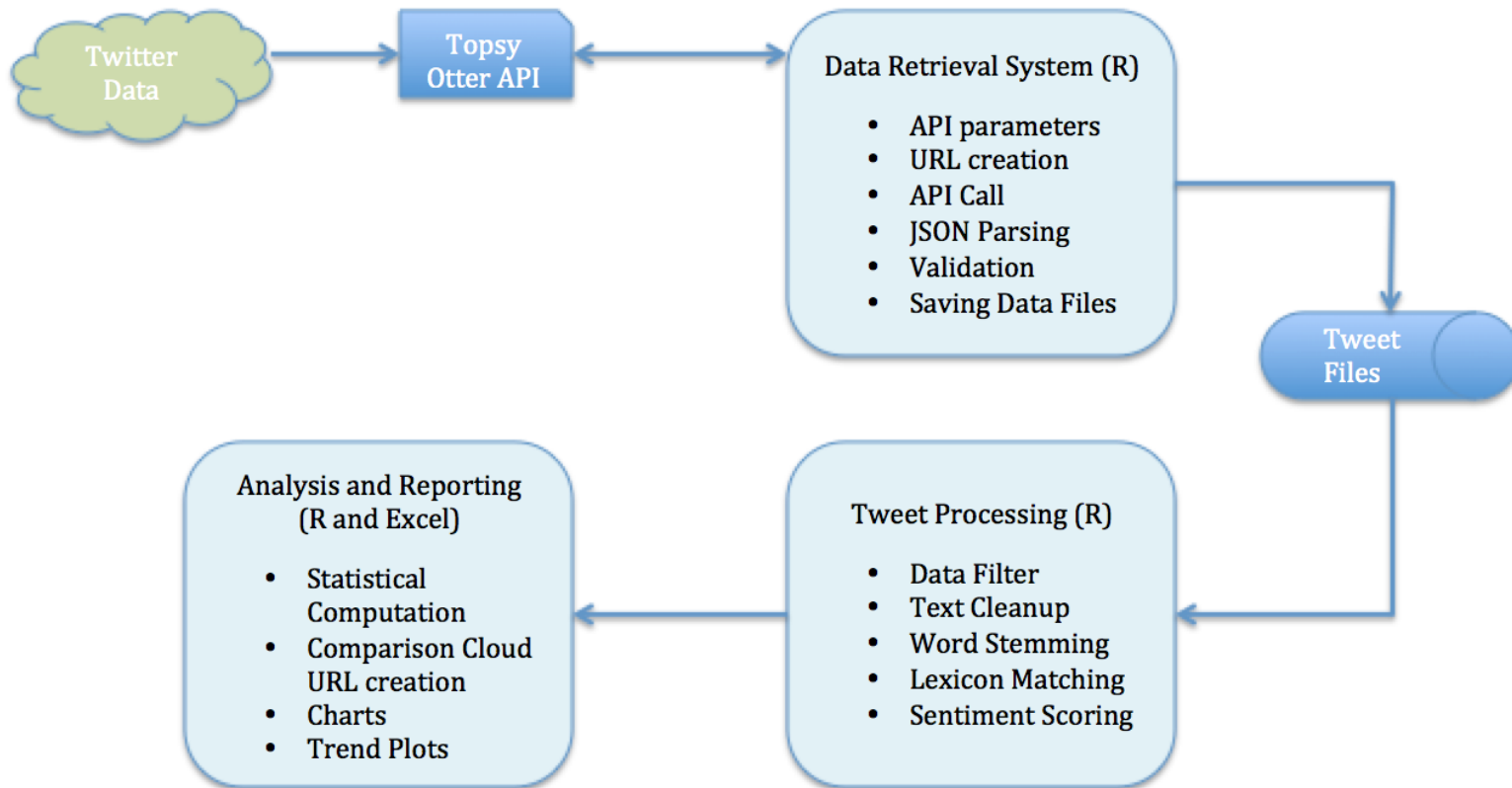
Sentiment Analysis

- What is “sentiment analysis”?
 - Using natural language processing, statistics, or machine learning methods to extract, identify, or otherwise characterize the sentiment content of a text unit
- Some examples
 - Is the product review positive or negative?
 - Have the bloggers’ attitudes about the stock changed since the acquisition?
 - How are people responding to the government’s post/news item on Twitter?
 - Other possible tasks: question answering, summarization
- Sentiment analysis in business, politics, law/policy making, sociology, and psychology

Limitations & Challenges

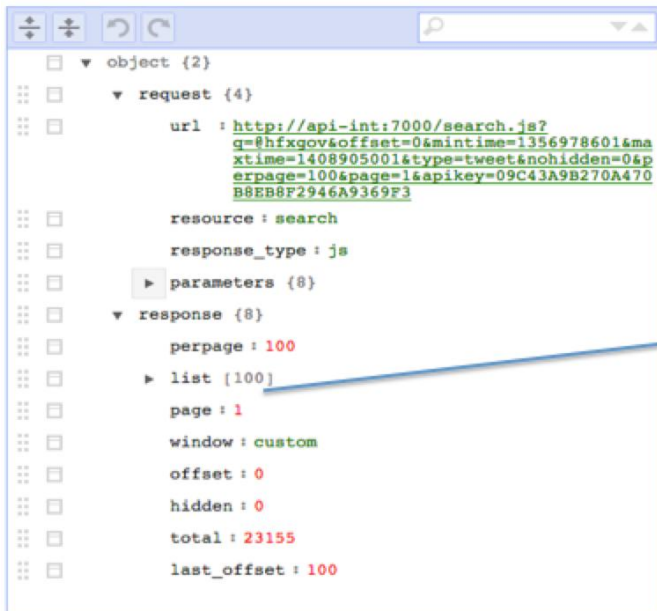
- Due to fair usage policy of web API's, all tweets were not available.
- Twitter search and streaming API's could not be used: limitations of 100 top tweets only (i.e., historical tweets were not available)
- Use topsy.com as an alternative: lists top 1000 tweets from historical data
- A programming looping logic was applied to repeat the retrieval process
- Limitations of topsy.com: re-tweet counts, geographic locations, etc. were not available

Methodology



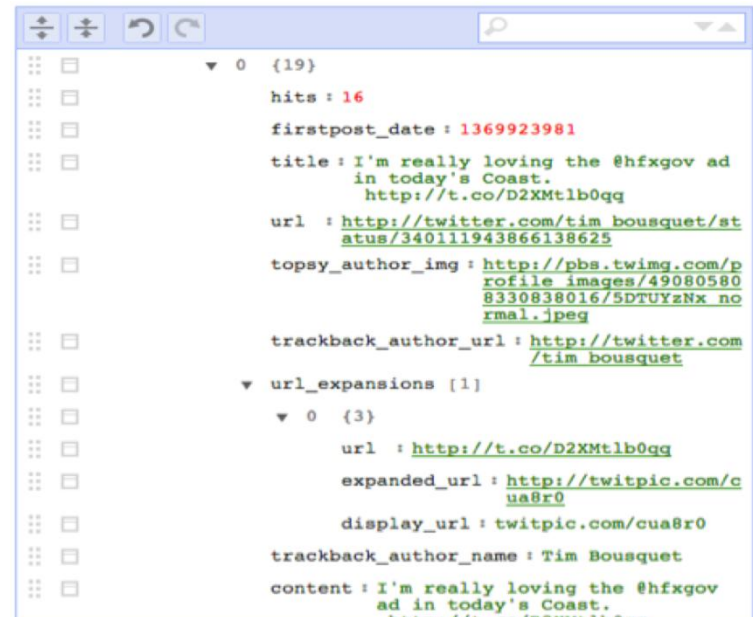
Methodology: Data Collection

- Topsy API was used to retrieve the tweets
 - An API URL example:
<http://otter.topsy.com/search.js?q=@hfxgov&offset=0&mintime=1356978601&maxtime=1408905001&type=tweet&nohidden=0&perpage=100&page=1&apikey=09C43A9B270A470B8EB8F2946A9369F3>
- A batch script in R was executed to retrieve these tweets
- The API response: a JSON data file (a tree/XML like format)



```
object {2}
  request {4}
    url : http://api-int:7000/search.js?q=@hfxgov&offset=0&mintime=1356978601&maxtime=1408905001&type=tweet&nohidden=0&perpage=100&page=1&apikey=09C43A9B270A470B8EB8F2946A9369F3
    resource : search
    response_type : js
    parameters {8}
  response {8}
    perpage : 100
    list [100]
    page : 1
    window : custom
    offset : 0
    hidden : 0
    total : 23155
    last_offset : 100
```

List of 100 Tweets.



```
{19}
  hits : 16
  firstpost_date : 1369923981
  title : I'm really loving the @hfxgov ad in today's Coast.
  url : http://twitter.com/tim_bousquet/status/340111943866138625
  topsy_author_img : http://pbs.twimg.com/profile_images/490805808330838016/5DTUYzNx_no_rmal.jpeg
  traceback_author_url : http://twitter.com/tim_bousquet
  url_expansions [1]
    {3}
      url : http://t.co/D2XMtlb0qq
      expanded_url : http://twitpic.com/cua8r0
      display_url : twitpic.com/cua8r0
      traceback_author_name : Tim Bousquet
      content : I'm really loving the @hfxgov ad in today's Coast.
        http://t.co/D2XMtlb0qq
```

Methodology: Data Preparation

- The retrieved data was cleansed by removing:
 - symbols
 - punctuations
 - special characters
 - URLs
 - numbers

Methodology: Sentiment Analysis

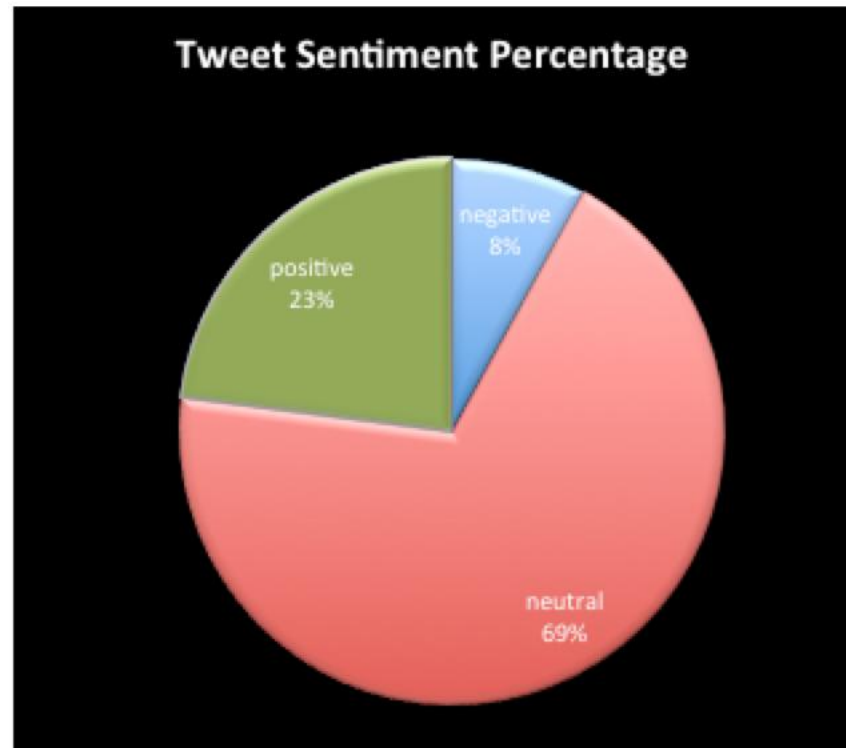
- Bag of Words approach was used for sentiment analysis.
 - Stemming: Each tweet was stemmed into the group of English words
 - Matching: A match of each word was searched in the lexicon database (total 6135 words in the lexicon; 2230 positive and 3905 negative)

abidance	positive	abandoned	negative
abidance	positive	abandonmer	negative
abilities	positive	aberration	negative
ability	positive	aberration	negative
able	positive	abhorred	negative
above	positive	abhorrence	negative
above-average	positive	abhorrent	negative
abundant	positive	abhorrently	negative

- Scoring: Positive and negative matches were summed to define a score of each tweet
- Polarity: $(P-N)/(P+N)$, where P=total sum of positive sentiment words; N=total sum of negative sentiment words
- Results were grouped and combined on a monthly basis.

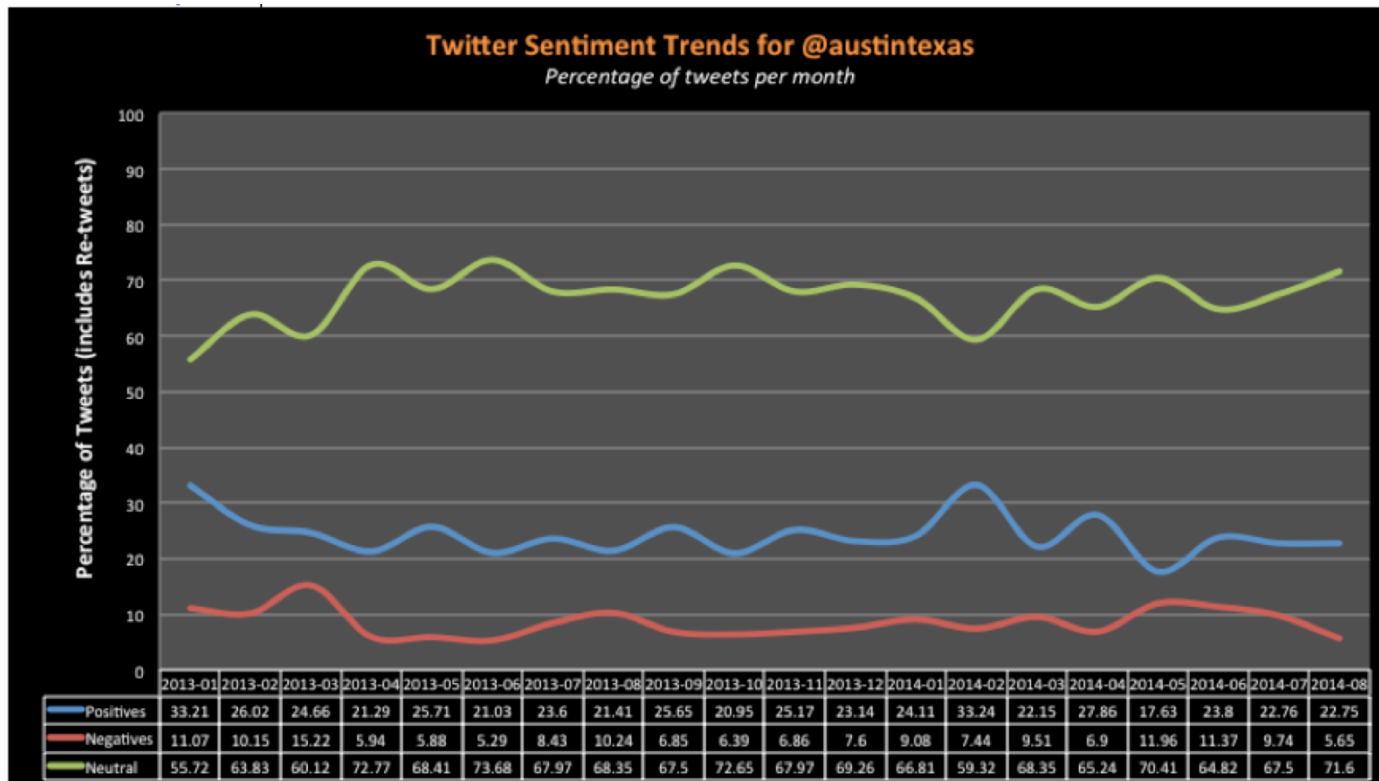
Analysis Results (@austintexasgov)

- Overall sentiment classification



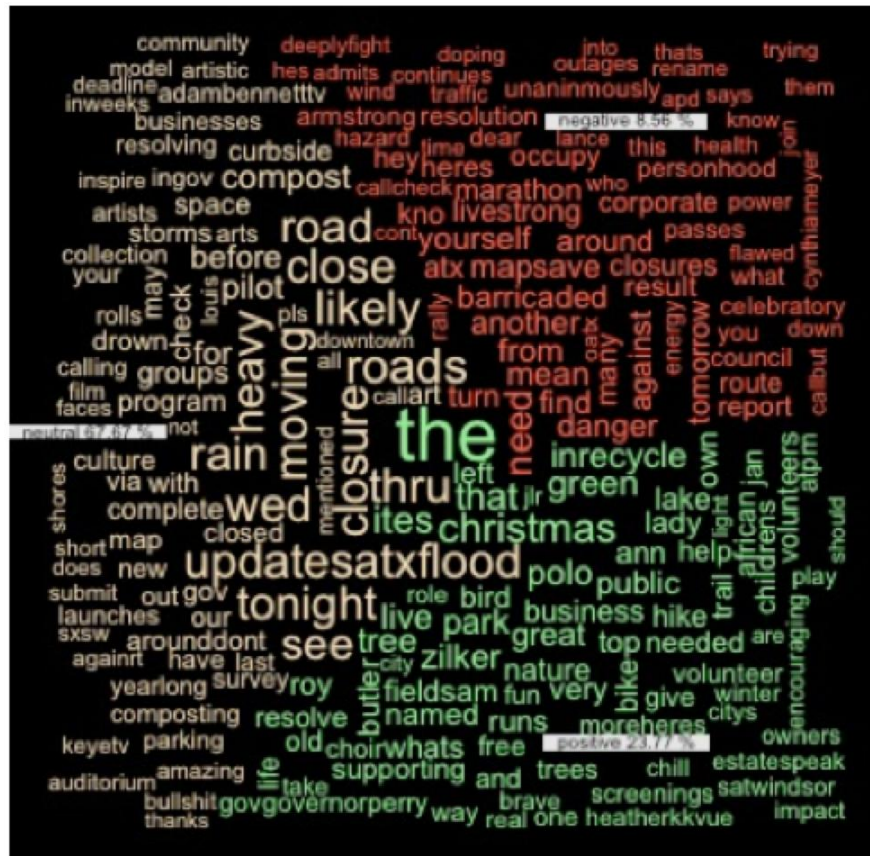
Analysis Results (@austintexasgov)

- Sentiment analysis plot



Analysis Results (@austintexasgov)

- Word cloud



Research Project #2

Quantitative Analytics for Library User Engagement Strategies through Social Media: Pinterest and Twitter (with Zou and Dey)

- Among many social media platforms, Pinterest is a new visual discovery social medium in which people can upload images and then collect ideas coming from different users with the same interests.
- User feedback can be considered a significant resource for libraries to customize their services to better engage their users.

Research Project #2 (cont.)

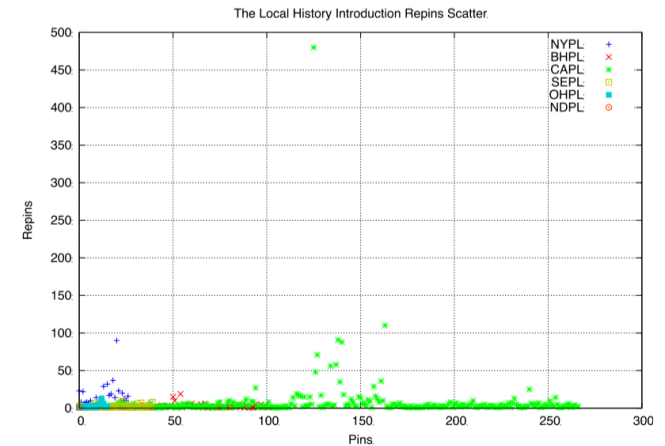
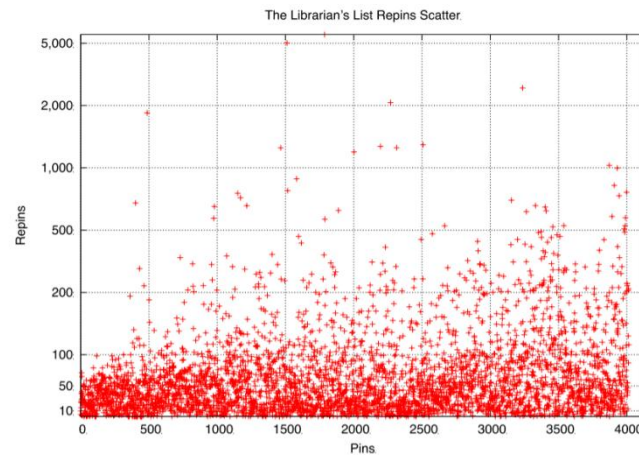
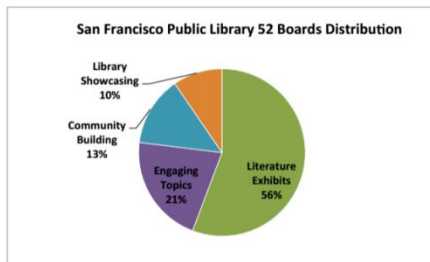
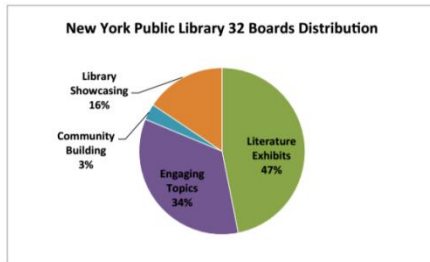
Quantitative Analytics for Library User Engagement Strategies through Social Media: Pinterest and Twitter (with Zou and Dey)

- The “20 Great Ways Libraries Are Using Pinterest Right Now” (Dunn 2012)
- Categorize 20 user engagement methods into four categories:
 - Literature exhibits
 - Engaging topic
 - Community building
 - Library showcasing
- 10 selected libraries were studied: NYPL, SJPL, SFPL, LAPL, BHPL, CAPL, SEPL, HTPL, OHPL, NDPL

Research Project #2 (cont.)

Quantitative Analytics for Library User Engagement Strategies through Social Media: Pinterest and Twitter (with Zou and Dey)

- Pinterest metrics
 - Pin, board, followers, following, repin, like, comment
- Preliminary findings



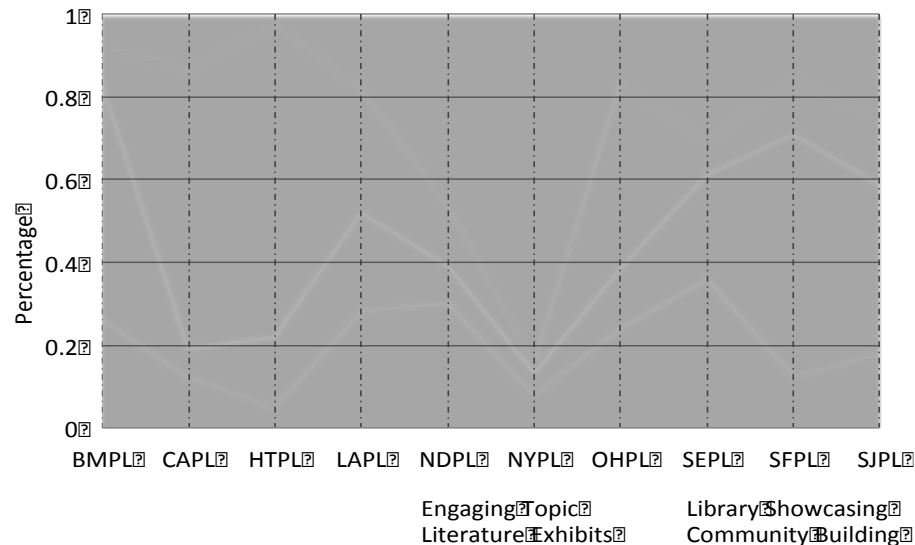
Research Project #2 (cont.)

Quantitative Analytics for Library User Engagement Strategies through Social Media: Pinterest and Twitter (with Zou and Dey)

- Twitter analysis: topic modeling
 - Looks for patterns in the use of words and injects semantic meaning into vocabulary; a topic consists of a cluster of words that frequently occur together
 - “MALLET” was adopted

Literature Exhibits
archives
books
list
read
bestselling
articles
cookbook
benefit
photo
cover

- Results



Useful Tools

- R (check out *R Programming* on Coursera!)
- Splunk
- Data mining techniques (check out <http://www.kdnuggets.com/>)
- Data visualizations (check out Many Eyes: <http://www-969.ibm.com/software/analytics/manyeyes/>)

Conclusions

- Social media analytics for...
 - Capturing user data to understand attitudes, opinions and trends, and manage online reputation
 - Predicting customer behavior and improve customer satisfaction by anticipating customer needs and recommending next best actions
 - Creating customized campaigns that resonate with social media participants
 - Identifying the primary influencers within specific social network channels