IIT School of Applied Technology

ILLINOIS INSTITUTE OF TECHNOLOGY

information technology & management

529 Advanced Data Analytics

October 25, 27 2016

Week 10 Presentation

Week 10 Topic: Agenda

Sentiment Analysis of Tweets:

- Week 10:
    - Create a Dev account on Twitter
    - Store tweets in corpus/DF
    - Develop own sentiment scoring function
- Week 11: Visualize analytics in Tableau
- Week 12: Store tweets in Hadoop
- Week 13: Store tweets in MongoDB

- Final Exam update

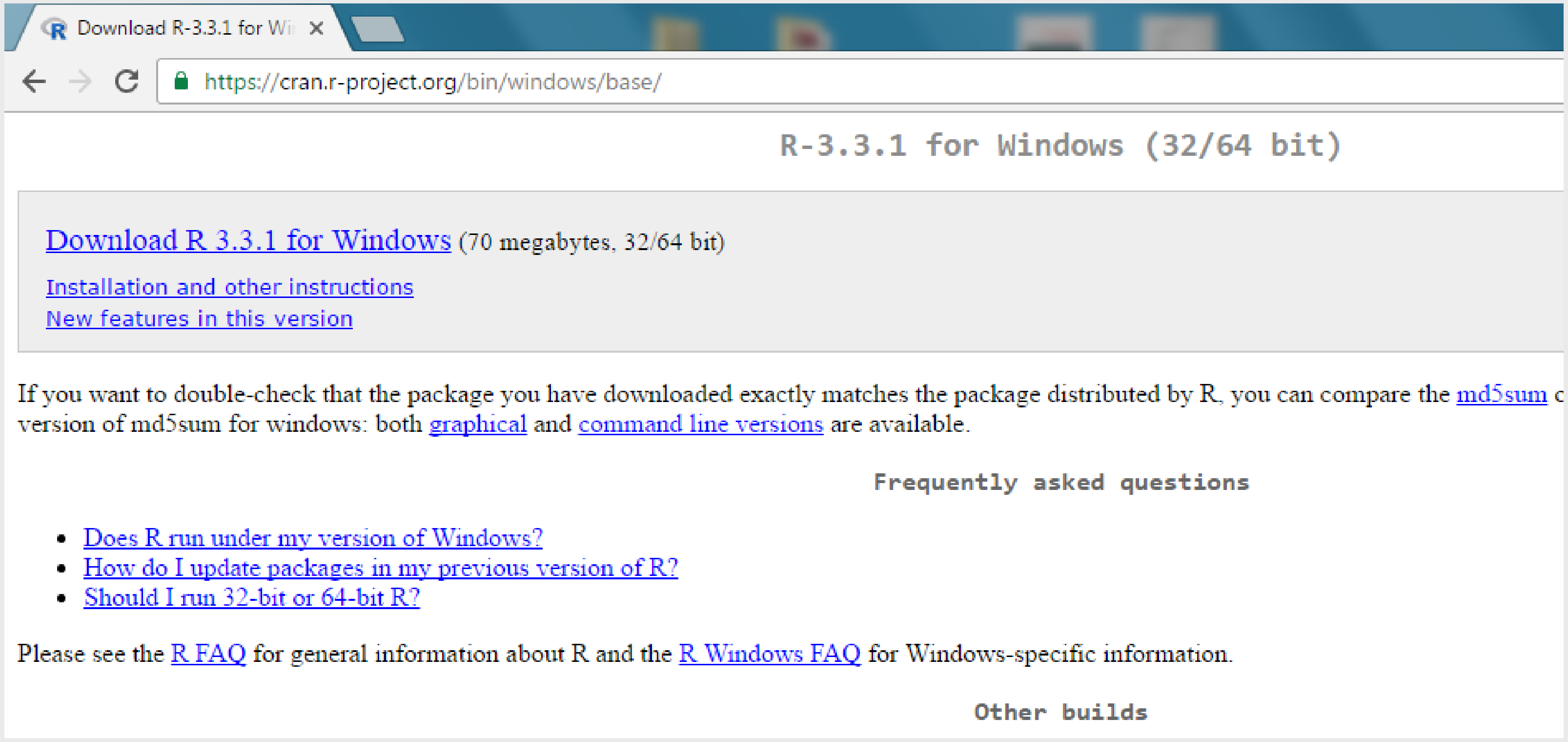
Week 10 Topic: Reference blogs/links

Use following examples as reference for the coming weeks:

- Donald Trump: https://www.r-bloggers.com/sentiment-analysis-on-donald-trump-using-r-and-tableau/
- Tweets to MongoDB: https://www.r-bloggers.com/gathering-twitter-data-with-the-twitter2mongo-package/
- Super Tuesday: https://www.r-bloggers.com/how-to-use-r-to-scrape-tweets-super-tuesday-2016/
- Jazz: https://jazzanalytics.wordpress.com/2016/09/12/sentiment-analysis-on-narendra-modi-using-r/
- Basic Sentiment Example: https://sites.google.com/site/miningtwitter/questions/sentiment/analysis

Week 10 Topic: R Version Needed

- Update your version of R to 3.2.5 or higher.
- Check your current version with ***R.Version()***


Download R-3.3.1 for Wi

https://cran.r-project.org/bin/windows/base/

R-3.3.1 for Windows (32/64 bit)

Download R 3.3.1 for Windows (70 megabytes, 32/64 bit)

Installation and other instructions
New features in this version

If you want to double-check that the package you have downloaded exactly matches the package distributed by R, you can compare the md5sum c
version of md5sum for windows: both graphical and command line versions are available.

Frequently asked questions

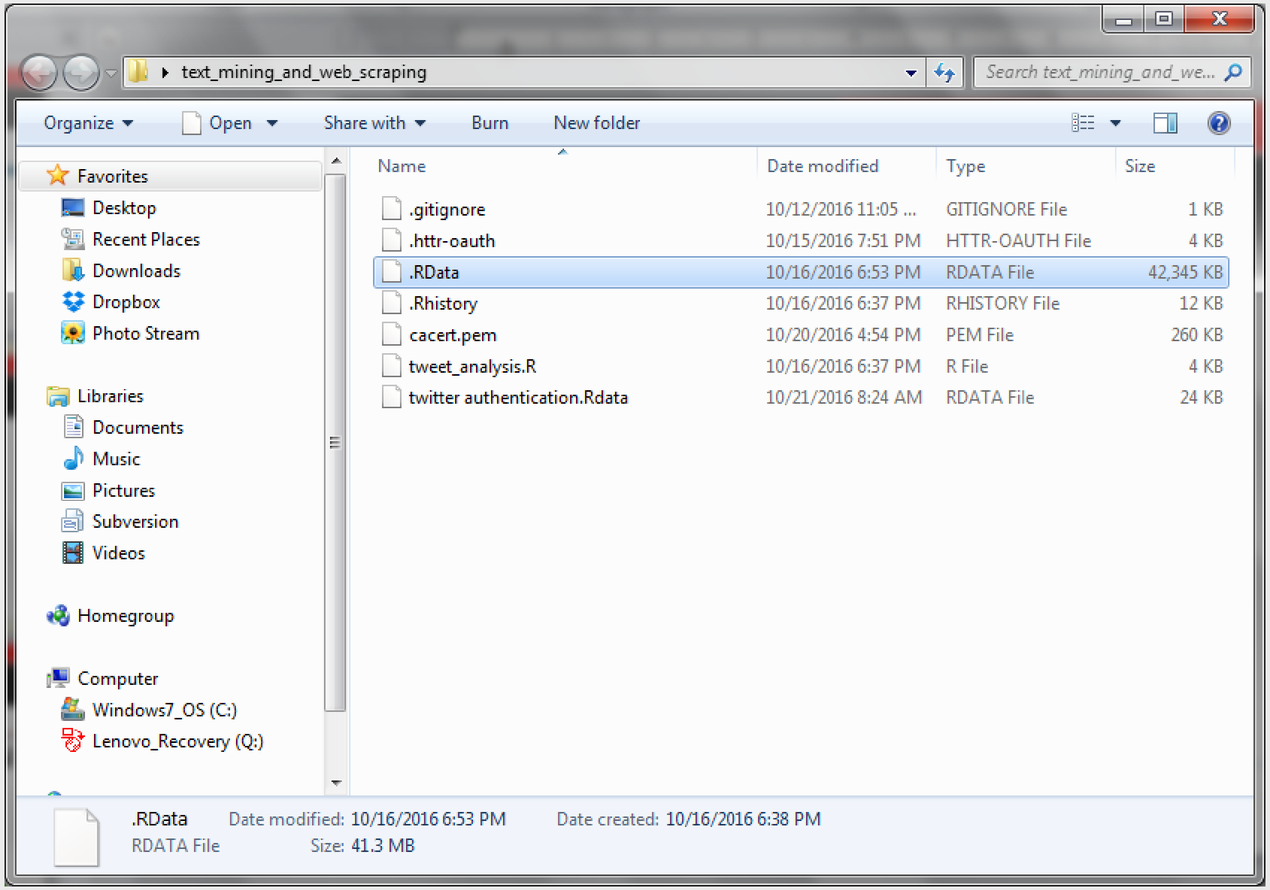
- Does R run under my version of Windows?
- How do I update packages in my previous version of R?
- Should I run 32-bit or 64-bit R?

Please see the R FAQ for general information about R and the R Windows FAQ for Windows-specific information.

Other builds

Week 10 Topic:
Set up your working directory

- Check your working directory with ***getwd()***
- Set your working directory with ***setwd("C:/Users/~~~~")***



text_mining_and_web_scraping

Name	Date modified	Type	Size
.gitignore	10/12/2016 11:05 ...	GITIGNORE File	1 KB
.httr-oauth	10/15/2016 7:51 PM	HTTR-OAUTH File	4 KB
.RData	10/16/2016 6:53 PM	RDATA File	42,345 KB
.Rhistory	10/16/2016 6:37 PM	RHISTORY File	12 KB
cacert.pem	10/20/2016 4:54 PM	PEM File	260 KB
tweet_analysis.R	10/16/2016 6:37 PM	R File	4 KB
twitter authentication.Rdata	10/21/2016 8:24 AM	RDATA File	24 KB

.RData Date modified: 10/16/2016 6:53 PM Date created: 10/16/2016 6:38 PM
RDATA File Size: 41.3 MB

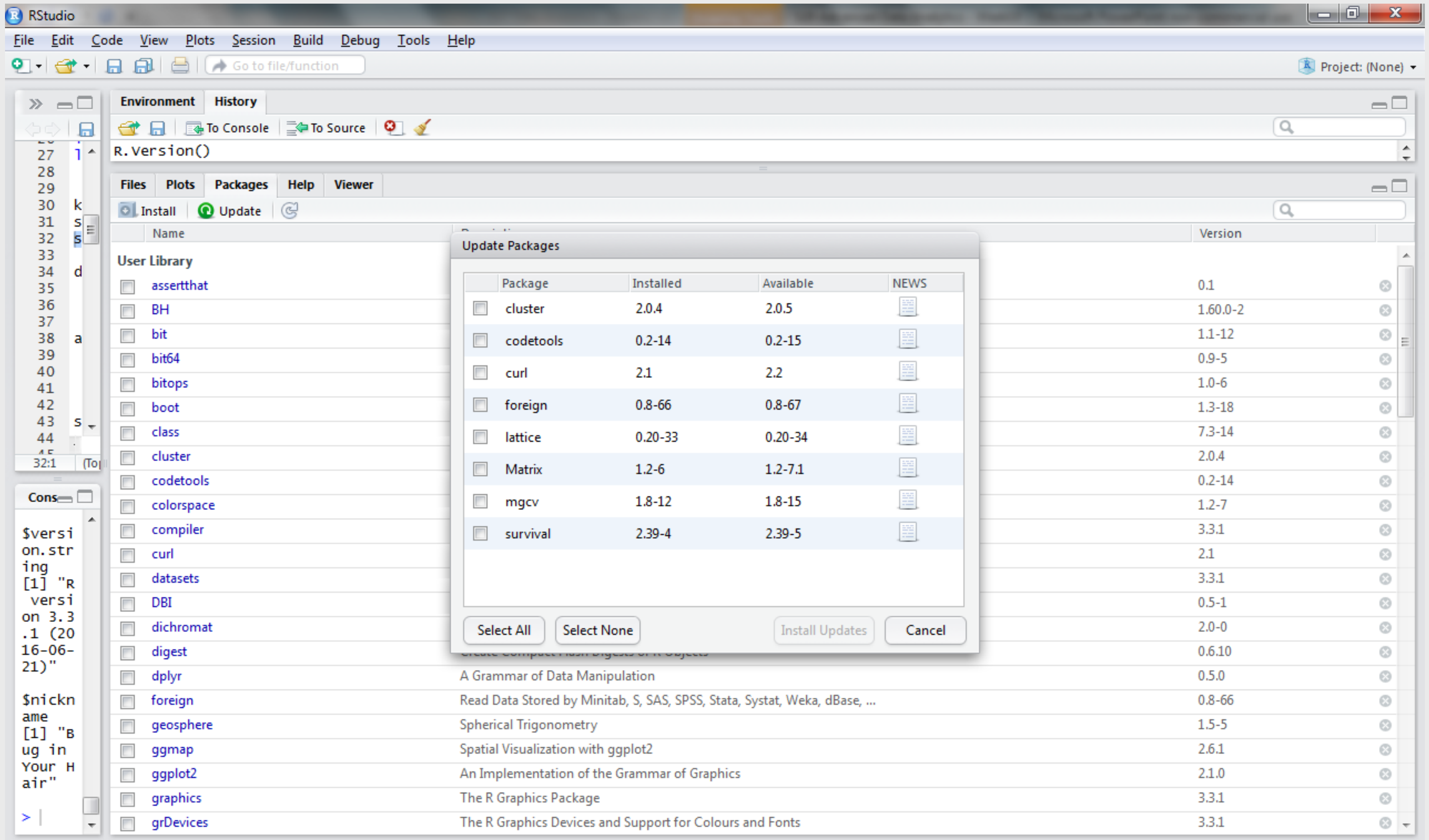
Week 10 Topic: R Packages/Libraries Needed

- install.packages("twitteR")
- install.packages("httpuv")
- install.packages("ROAuth")
- install.packages("stringr")
- install.packages("tm")
- install.packages("ggmap")
- install.packages("dplyr")
- install.packages("plyr")
- install.packages("wordcloud")
- install.packages("openssl")

- library(twitteR)
- library(httpuv)
- library(ROAuth)
- library(RCurl)
- library(stringr)
- library(tm)
- library(ggmap)
- library(plyr)
- library(dplyr)
- library(wordcloud)
- library(openssl)

Week 10 Topic:
Update R Packages/Libraries

- If you have packages installed previously, you can update packages:



```
R.version()
```

```
$versi
on.str
ing
[1] "R
 versi
on 3.3
.1 (20
16-06-
21)"

$nickn
ame
[1] "B
ug in
Your H
air"
```

Update Packages

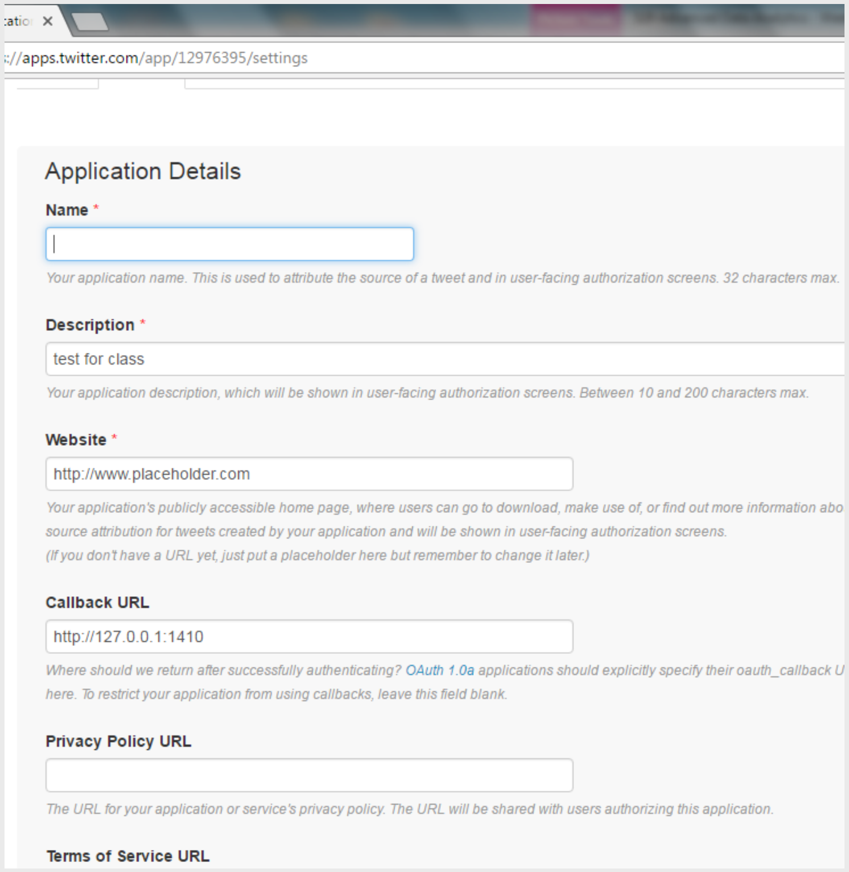
	Package	Installed	Available	NEWS
☐	cluster	2.0.4	2.0.5	
☐	codetools	0.2-14	0.2-15	
☐	curl	2.1	2.2	
☐	foreign	0.8-66	0.8-67	
☐	lattice	0.20-33	0.20-34	
☐	Matrix	1.2-6	1.2-7.1	
☐	mgcv	1.8-12	1.8-15	
☐	survival	2.39-4	2.39-5	

User Library

Name	Description	Version
assertthat		0.1
BH		1.60.0-2
bit		1.1-12
bit64		0.9-5
bitops		1.0-6
boot		1.3-18
class		7.3-14
cluster		2.0.4
codetools		0.2-14
colorspace		1.2-7
compiler		3.3.1
curl		2.1
datasets		3.3.1
DBI		0.5-1
dichromat		2.0-0
digest	Create Compact Hash Digests of R Objects	0.6.10
dplyr	A Grammar of Data Manipulation	0.5.0
foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
geosphere	Spherical Trigonometry	1.5-5
ggmap	Spatial Visualization with ggplot2	2.6.1
ggplot2	An Implementation of the Grammar of Graphics	2.1.0
graphics	The R Graphics Package	3.3.1
grDevices	The R Graphics Devices and Support for Colours and Fonts	3.3.1

Week 10 Topic: Twitter Account/App set up

- In order to have access to Twitter data programmatically, one needs to create an app that interacts with the Twitter API.
- The first step is the registration of your app. In particular, you need to point your browser to ***https://apps.twitter.com/*** log-in to Twitter with a phone number (if you're not already logged in) and register a new application.
- Add the name and description of your app along with a website name. The website can be a test website.
- There's also a field for callback URL, but that's optional.



://apps.twitter.com/app/12976395/settings

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 *

test for class

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

Website *

http://www.placeholder.com

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information abo
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

http://127.0.0.1:1410

Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback U
here. To restrict your application from using callbacks, leave this field blank.

Privacy Policy URL

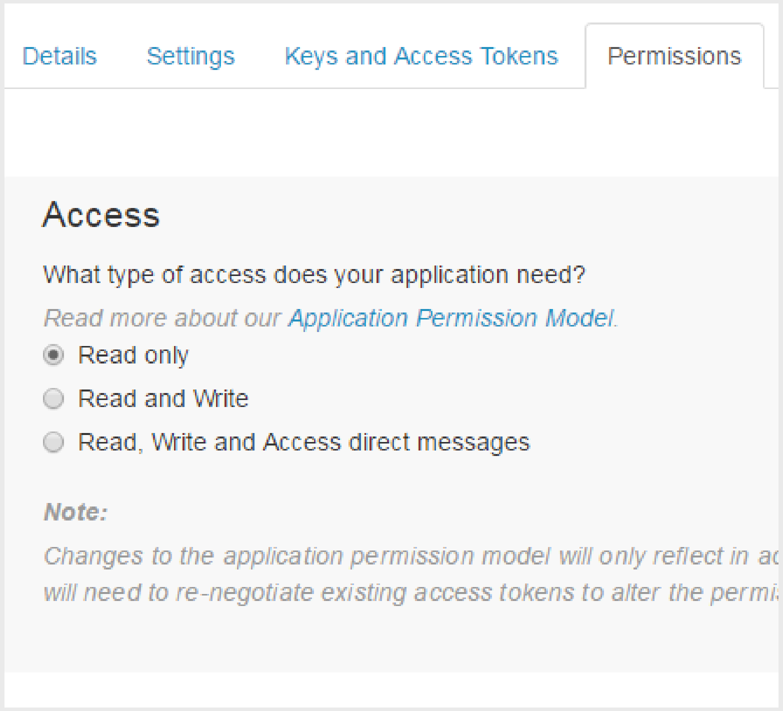
The URL for your application or service's privacy policy. The URL will be shared with users authorizing this application.

Terms of Service URL

ITM - 529

Week 10 Topic: Twitter Account/App set up (cont.)

- You will receive a *consumer key* and a *consumer secret*: these are application settings that should always be kept private.
- From the configuration page of your app, you can also require an access token and an access token secret. Similarly to the consumer keys, these strings must also be kept private: they provide the application access to Twitter on behalf of your account.
- The default permissions are read-only, which is all we need in our case, but if you decide to change your permission to provide writing features in your app, you must negotiate a new access token



Access

What type of access does your application need?

Read more about our Application Permission Model.

- ● Read only
- ○ Read and Write
- ○ Read, Write and Access direct messages

Note:

Changes to the application permission model will only reflect in ac
will need to re-negotiate existing access tokens to alter the permi

Week 10 Topic:
Authenticate App w Twitter

Grab your API keys and access tokens from Twitter:

```
>setup_twitter_oauth(api_key, api_secret, access_token, access_token_secret)
```

Or

```
>authenticate <- OAuthFactory$new(
consumerKey=key, consumerSecret=secret,
requestURL="https://api.twitter.com/oauth/request_token",
accessURL="https://api.twitter.com/oauth/access_token",
authURL="https://api.twitter.com/oauth/authorize")
>setup_twitter_oauth(key, secret)
```

Save authentication:

```
>download.file(url="http://curl.haxx.se/ca/cacert.pem",
        destfile="C:/Users/sshin/Desktop/text_mining_and_web_scraping/cacert.pem",
        method="auto")
>save(authenticate, file="twitter authentication.Rdata")
```

Week 10 Topic:
Grab Tweets, Create Library, Store

- # Grab latest tweets for Donal Trump and Hillary Clinton:

tweets_trump <- searchTwitter('@realDonaldTrump', n=1500)

tweets_clinton <- searchTwitter('@HillaryClinton', n=1500)

- Loop over tweets and extract text library(plyr):

feed_trump = laply(tweets_trump, function(t) t$getText())

feed_clinton= laply(tweets_clinton, function(t) t$getText())

- Write to csv as needed:

write.csv(feed_trump, "donaldtrump.csv",row.names = F)
write.csv(feed_clinton, "hilarlaryclinton.csv",row.names = F)

Week 10 Topic: **Cleanse Tweets**

Now you've got a bunch of text data for Trump and Clinton, so how do we decide what's a "good" tweet and a "bad" tweet? This is where we turned to the Hu and Liu Opinion Lexicon, a list of 6800 positive and negative words compiled by Bing Liu and Minqing Hu of the University of Illinois at Chicago.

- Unpack the Opinion Lexicon into your working directory.

```
# Read in dictionary of positive and negative works
yay = scan('opinion-lexicon-English/positive-words.txt', what='character', comment.char=';')
boo = scan('opinion-lexicon-English/negative-words.txt', what='character', comment.char=';')

# Add a few twitter-specific negative phrases
bad_text = c(boo, 'wtf', 'epicfail', 'douchebag')
good_text = c(yay, 'upgrade', ':)', '#iVoted', 'voted')
```

Week 10 Topic: **Scoring function**

- Now, you’ve got your list of tweets and your list of opinionated words. The next thing to do is score the text of the tweets compared to how many of the “bad” and “good” words show up in each.
- For this we’ll need a giant R function filled with lots of good gsub and match functions. Thanks to Jeff Breen for the function on which this was based:

```
score.sentiment = function(sentences, good_text, bad_text, .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 of scores back, so we use
# "l" + "a" + "ply" = "laply":
, text=sentences) return(scores.df) }
```

Week 10 Topic: **Scoring function (cont.)**

```
scores = laply(sentences, function(sentence, good_text, bad_text) {

# clean up sentences with R's regex-driven global substitute, gsub():
sentence = gsub('[[:punct:]]', '', sentence)
sentence = gsub('[[:cntrl:]]', '', sentence)
sentence = gsub('\\d+', '', sentence)

#to remove emojis
sentence <- iconv(sentence, 'UTF-8', 'ASCII')

# and convert to lower case:
sentence = tolower(sentence)
```

Week 10 Topic:
Scoring function (cont.)

```
# 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, 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)
) }
```

Week 10 Topic:
Scoring function (cont.)

```
# 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) }
```

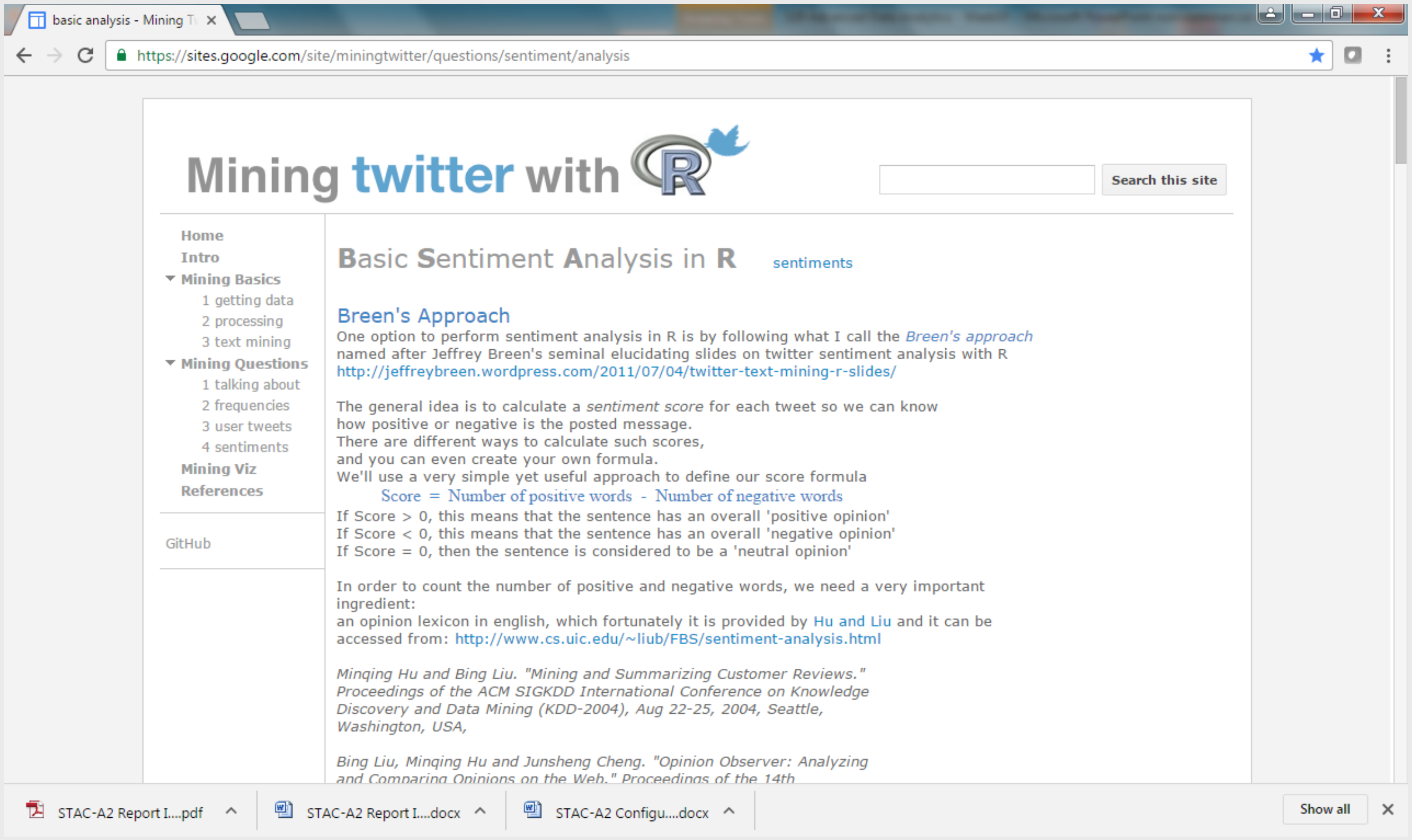
Week 10 Topic: **Calling and Plotting**

```
# Call the function and return a data frame
feelthatrump <- score.sentiment(feed_trump, good_text, bad_text, .progress='text')
feelthaclinton <- score.sentiment(feed_clinton, good_text, bad_text, .progress='text')

# Nice little quick plot
qplot(factor(score), data=feelthatrump, geom="bar", xlab = "Sentiment Score")
qplot(factor(score), data=feelthaclinton, geom="bar", xlab = "Sentiment Score")
```

Week 10 Topic: Reference

- https://sites.google.com/site/miningtwitter/questions/sentiment/analysis:



Basic Sentiment Analysis in R sentiments

Breen's Approach

One option to perform sentiment analysis in R is by following what I call the *Breen's approach* named after Jeffrey Breen's seminal elucidating slides on twitter sentiment analysis with R
http://jeffreybreen.wordpress.com/2011/07/04/twitter-text-mining-r-slides/

The general idea is to calculate a *sentiment score* for each tweet so we can know how positive or negative is the posted message.
There are different ways to calculate such scores, and you can even create your own formula.
We'll use a very simple yet useful approach to define our score formula

$$\text{Score} = \text{Number of positive words} - \text{Number of negative words}$$

If Score > 0, this means that the sentence has an overall 'positive opinion'
If Score < 0, this means that the sentence has an overall 'negative opinion'
If Score = 0, then the sentence is considered to be a 'neutral opinion'

In order to count the number of positive and negative words, we need a very important ingredient:
an opinion lexicon in english, which fortunately it is provided by Hu and Liu and it can be accessed from: http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews." Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA,

Bing Liu, Minqing Hu and Junsheng Cheng. "Opinion Observer: Analyzing and Comparing Opinions on the Web." Proceedings of the 14th

ITM - 529

Week 10 Topic:
Week 10 Assignment

Develop your own sentiment analysis scoring function:

1) Customize/Augment/Update the “good text”, “bad text” repository with own research/list of words. Survey the tweets (in XLS or other) and determine words to be considered for either. Submit a table of “good” and “bad” words.
2) Determine additional levels of scoring e.g., more than just good and bad, weighting of words by importance, weighting of words by frequency etc. Submit scoring logic in a table.
3) Develop final scoring method and question. Submit the final scoring equation and function.
4) Share in discussion topic