Text Mining with R *

Yanchang Zhao

http://www.RDataMining.com

R and Data Mining Course

Beijing University of Posts and Telecommunications,

Beijing, China

July 2019

---

*Chapter 10: Text Mining, in *R and Data Mining: Examples and Case Studies*.
http://www.rdatamining.com/docs/RDataMining-book.pdf

Contents

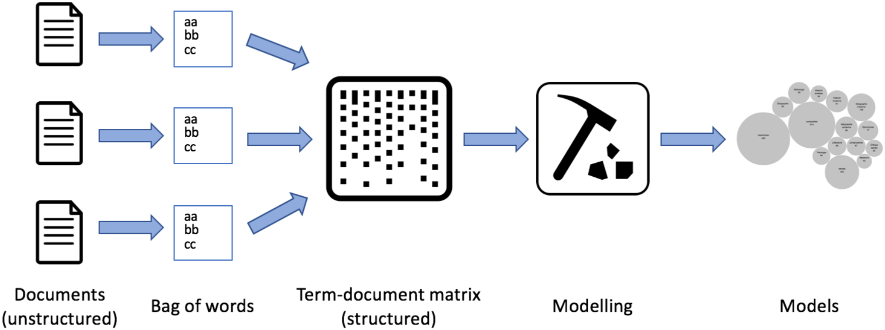
Text Data

- Text documents in a natural language
- Unstructured
- Documents in plain text, Word or PDF format
- Emails, online chat logs and phone transcripts
- Online news and forums, blogs, micro-blogs and social media
- ...

Typical Process of Text Mining

1. Transform text into structured data
   - Term-Document Matrix (TDM)
   - Entities and relations
   - ...
2. Apply traditional data mining techniques to the above structured data
   - Clustering
   - Classification
   - Social Network Analysis (SNA)
   - ...

Typical Process of Text Mining (cont.)



Term-Document Matrix (TDM)

- Also known as Document-Term Matrix (DTM)
- A 2D matrix
- Rows: terms or words
- Columns: documents
- Entry $m_{i,j}$: number of occurrences of term t_i in document d_j
- Term weighting schemes: Term Frequency, Binary Weight, TF-IDF, etc.

TF-IDF

- Term Frequency (TF) $\mathrm{tf}_{i,j}$: the number of occurrences of term t_i in document d_j
- Inverse Document Frequency (IDF) for term t_i is:

$$\mathrm{idf}_i = \log_2 \frac{|D|}{|\{d \mid t_i \in d\}|} \tag{1}$$

  $|D|$: the total number of documents
  $|\{d \mid t_i \in d\}|$: the number of documents where term t_i appears
- Term Frequency - Inverse Document Frequency (TF-IDF)

$$\mathrm{tfidf} = \mathrm{tf}_{i,j} \cdot \mathrm{idf}_i \tag{2}$$

- IDF reduces the weight of terms that occur frequently in documents and increases the weight of terms that occur rarely.

An Example of TDM

Doc1: I like R.
Doc2: I like Python.

Term Frequency

	Doc1	Doc2
I	1	1
like	1	1
Python	0	1
R	1	0

IDF

	IDF
I	0
like	0
Python	1
R	1

TF-IDF

	Doc1	Doc2
I	0	0
like	0	0
Python	0	1
R	1	0

An Example of TDM

Doc1: I like R.
Doc2: I like Python.

Term Frequency

	Doc1	Doc2
I	1	1
like	1	1
Python	0	1
R	1	0

IDF

	IDF
I	0
like	0
Python	1
R	1

TF-IDF

	Doc1	Doc2
I	0	0
like	0	0
Python	0	1
R	1	0

Terms that can distinguish different documents are given greater weights.

An Example of TDM (cont.)

Doc1: I like R.
Doc2: I like Python.

Term Frequency

	Doc1	Doc2
I	1	1
like	1	1
Python	0	1
R	1	0

IDF

	IDF
I	0
like	0
Python	1
R	1

Normalized Term Frequency

	Doc1	Doc2
I	0.33	0.33
like	0.33	0.33
Python	0	0.33
R	0.33	0

Normalized TF-IDF

	Doc1	Doc2
I	0	0
like	0	0
Python	0	0.33
R	0.33	0

An Example of Term Weighting in R

```r
## term weighting
library(magrittr)
library(tm) ## package for text mining
a <- c("I like R", "I like Python")
## build corpus
b <- a %>% VectorSource() %>% Corpus()
## build term document matrix
m <- b %>% TermDocumentMatrix(control=list(wordLengths=c(1, Inf)))
m %>% inspect()

## various term weighting schemes
m %>% weightBin() %>% inspect() ## binary weighting
m %>% weightTf() %>% inspect() ## term frequency
m %>% weightTfIdf(normalize=F) %>% inspect() ## TF-IDF
m %>% weightTfIdf(normalize=T) %>% inspect() ## normalized TF-IDF
```

More options provided in package *tm*:

- `weightSMART`
- `WeightFunction`

Text Mining Tasks

- Text classification
- Text clustering and categorization
- Topic modelling
- Sentiment analysis
- Document summarization
- Entity and relation extraction
- ...

Topic Modelling

- To identify topics in a set of documents
- It groups both documents that use similar words and words that occur in a similar set of documents.
- Intuition: Documents related to R would contain more words like R, ggplot2, plyr, stringr, knitr and other R packages, than Python related keywords like Python, NumPy, SciPy, Matplotlib, etc.
- A document can be of multiple topics in different proportions. For instance, a document can be 90% about R and 10% about Python. $\Rightarrow$ soft/fuzzy clustering
- Latent Dirichlet Allocation (LDA): the most widely used topic model

Sentiment Analysis

- Also known as opinion mining
- To determine attitude, polarity or emotions from documents
- Polarity: positive, negative, netural
- Emotions: angry, sad, happy, bored, afraid, etc.
- Method:
  1. identify invidual words and phrases and map them to different emotional scales
  2. adjust the sentiment value of a concept based on modifications surrounding it

Document Summarization

- To create a summary with major points of the orignial document
- Approaches
  - Extraction: select a subset of existing words, phrases or sentences to build a summary
  - Abstraction: use natural language generation techniques to build a summary that is similar to natural language

Entity and Relationship Extraction

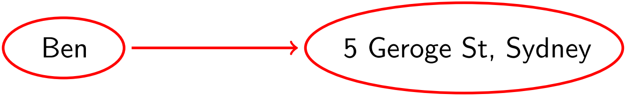
- Named Entity Recognition (NER): identify named entities in text into pre-defined categories, such as person names, organizations, locations, date and time, etc.
- Relationship Extraction: identify associations among entities
- Example:
  Ben lives at 5 Geroge St, Sydney.

Entity and Relationship Extraction

- Named Entity Recognition (NER): identify named entities in text into pre-defined categories, such as person names, organizations, locations, date and time, etc.
- Relationship Extraction: identify associations among entities
- Example:
  Ben lives at 5 Geroge St, Sydney.

Entity and Relationship Extraction

- Named Entity Recognition (NER): identify named entities in text into pre-defined categories, such as person names, organizations, locations, date and time, etc.
- Relationship Extraction: identify associations among entities
- Example:
  Ben lives at 5 Geroge St, Sydney.



Contents

Twitter

- An online social networking service that enables users to send and read short 280-character (used to be 140 before November 2017) messages called "tweets" (Wikipedia)
- Over 300 million monthly active users (as of 2018)
- Creating over 500 million tweets per day

RDataMining Twitter Account

Process†

1. Extract tweets and followers from the Twitter website with R and the *twitteR* package
2. With the *tm* package, clean text by removing punctuations, numbers, hyperlinks and stop words, followed by stemming and stem completion
3. Build a term-document matrix
4. Cluster Tweets with text clustering
5. Analyse topics with the *topicmodels* package
6. Analyse sentiment with the *sentiment140* package
7. Analyse following/followed and retweeting relationships with the *igraph* package

† More details in paper titled *Analysing Twitter Data with Text Mining and Social Network Analysis* [Zhao, 2013].

Retrieve Tweets

```
## Option 1: retrieve tweets from Twitter
library(twitteR)
library(ROAuth)
## Twitter authentication
setup_twitter_oauth(consumer_key, consumer_secret, access_token, access
## 3200 is the maximum to retrieve
tweets <- "RDataMining" %>% userTimeline(n = 3200)
```

See details of *Twitter Authentication with OAuth* in Section 3 of http://geoffjentry.hexdump.org/twitteR.pdf.

```
## Option 2: download @RDataMining tweets from RDataMining.com
library(twitteR)
url <- "http://www.rdatamining.com/data/RDataMining-Tweets-20160212.rds
download.file(url, destfile = "./data/RDataMining-Tweets-20160212.rds")
## load tweets into R
tweets <- readRDS("./data/RDataMining-Tweets-20160212.rds")
```

```r
(n.tweet <- tweets %>% length())
## [1] 448

# convert tweets to a data frame
tweets.df <- tweets %>% twListToDF()
# tweet #1
tweets.df[1, c("id", "created", "screenName", "replyToSN",
  "favoriteCount", "retweetCount", "longitude", "latitude", "text")]
##                   id             created  screenName replyToSN
## 1 697031245503418368 2016-02-09 12:16:13 RDataMining      <NA>
##   favoriteCount retweetCount longitude latitude
## 1            13           14        NA       NA
##                                                                ...
## 1 A Twitter dataset for text mining: @RDataMining Tweets ex...

# print tweet #1 and make text fit for slide width
tweets.df$text[1] %>% strwrap(60) %>% writeLines()
## A Twitter dataset for text mining: @RDataMining Tweets
## extracted on 3 February 2016. Download it at
## https://t.co/lQp94IvfPf
```

Text Cleaning Functions

- Convert to lower case: `tolower`
- Remove punctuation: `removePunctuation`
- Remove numbers: `removeNumbers`
- Remove URLs
- Remove stop words (like 'a', 'the', 'in'): `removeWords`, `stopwords`
- Remove extra white space: `stripWhitespace`

```
## text cleaning
library(tm)
# function for removing URLs, i.e.,
#  "http" followed by any non-space letters
removeURL <- function(x) gsub("http[^[:space:]]*", "", x)
# function for removing anything other than English letters or space
removeNumPunct <- function(x) gsub("[^[:alpha:][:space:]]*", "", x)
# customize stop words
myStopwords <- c(setdiff(stopwords('english'), c("r", "big")),
                 "use", "see", "used", "via", "amp")
```

See details of regular expressions by running ?regex in R console.

Text Cleaning

```r
# build a corpus and specify the source to be character vectors
corpus.raw <- tweets.df$text %>% VectorSource() %>% Corpus()

# text cleaning
corpus.cleaned <- corpus.raw %>%
  # convert to lower case
  tm_map(content_transformer(tolower)) %>%
  # remove URLs
  tm_map(content_transformer(removeURL)) %>%
  # remove numbers and punctuations
  tm_map(content_transformer(removeNumPunct)) %>%
  # remove stopwords
  tm_map(removeWords, myStopwords) %>%
  # remove extra whitespace
  tm_map(stripWhitespace)
```

Stemming and Stem Completion [‡]

```r
## stem words
corpus.stemmed <- corpus.cleaned %>% tm_map(stemDocument)

## stem completion
stemCompletion2 <- function(x, dictionary) {
  x <- unlist(strsplit(as.character(x), " "))
  x <- x[x != ""]
  x <- stemCompletion(x, dictionary=dictionary)
  x <- paste(x, sep="", collapse=" ")
  stripWhitespace(x)
}

corpus.completed <- corpus.stemmed %>%
  lapply(stemCompletion2, dictionary=corpus.cleaned) %>%
  VectorSource() %>% Corpus()
```

[‡] http://stackoverflow.com/questions/25206049/stemcompletion-is-not-working

Before/After Text Cleaning and Stemming

```
## compare text before/after cleaning
# original text
corpus.raw[[1]]$content %>% strwrap(60) %>% writeLines()
## A Twitter dataset for text mining: @RDataMining Tweets
## extracted on 3 February 2016. Download it at
## https://t.co/lQp94IvfPf

# after basic cleaning
corpus.cleaned[[1]]$content %>% strwrap(60) %>% writeLines()
## twitter dataset text mining rdatamining tweets extracted
## february download

# stemmed text
corpus.stemmed[[1]]$content %>% strwrap(60) %>% writeLines()
## twitter dataset text mine rdatamin tweet extract februari
## download

# after stem completion
corpus.completed[[1]]$content %>% strwrap(60) %>% writeLines()
## twitter dataset text miner rdatamining tweet extract
## download
```

Issues in Stem Completion: “Miner” vs “Mining”

```r
# count word frequence
wordFreq <- function(corpus, word) {
  results <- lapply(corpus,
    function(x) grep(as.character(x), pattern=paste0("\\<",word)) )
  sum(unlist(results))
}
n.miner <- corpus.cleaned %>% wordFreq("miner")
n.mining <- corpus.cleaned %>% wordFreq("mining")
cat(n.miner, n.mining)
```

```
## 9 104
```

```r
# replace old word with new word
replaceWord <- function(corpus, oldword, newword) {
  tm_map(corpus, content_transformer(gsub),
         pattern=oldword, replacement=newword)
}
corpus.completed <- corpus.completed %>%
  replaceWord("miner", "mining") %>%
  replaceWord("universidad", "university") %>%
  replaceWord("scienc", "science")
```

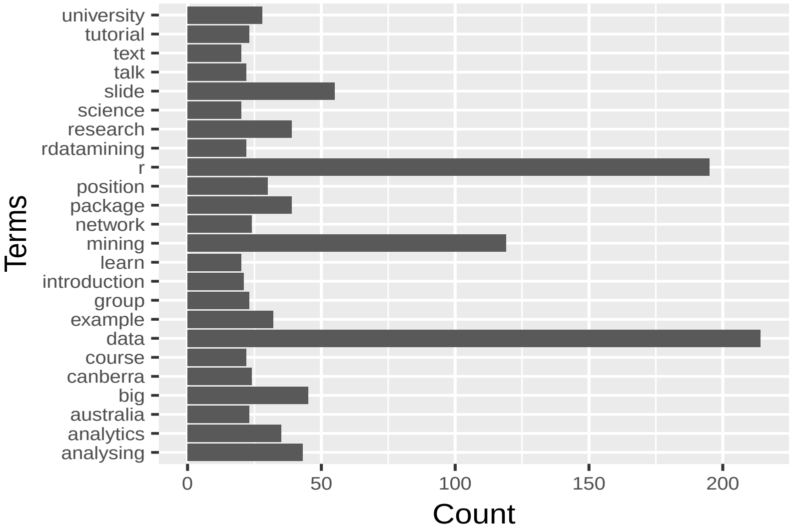
Build Term Document Matrix

```
## Build Term Document Matrix
tdm <- corpus.completed %>%
  TermDocumentMatrix(control = list(wordLengths = c(1, Inf)))  %>%
  print
## <<TermDocumentMatrix (terms: 1073, documents: 448)>>
## Non-/sparse entries: 3594/477110
## Sparsity           : 99%
## Maximal term length: 23
## Weighting          : term frequency (tf)

idx <- which(dimnames(tdm)$Terms %in% c("r", "data", "mining"))
tdm[idx, 21:30] %>% as.matrix()
##           Docs
## Terms     21 22 23 24 25 26 27 28 29 30
##   mining   0  0  0  0  1  0  0  0  0  1
##   data     0  1  0  0  1  0  0  0  0  1
##   r        1  1  1  1  0  1  0  1  1  1
```

Top Frequent Terms

```
# inspect frequent words
freq.terms <- tdm %>% findFreqTerms(lowfreq = 20) %>% print
##  [1] "mining"       "rdatamining"  "text"         "analytics"
##  [5] "australia"    "data"         "canberra"     "group"
##  [9] "university"   "science"      "slide"        "tutorial"
## [13] "big"          "learn"        "package"      "r"
## [17] "network"      "course"       "introduction" "talk"
## [21] "analysing"    "research"     "position"     "example"

term.freq <- tdm %>% as.matrix() %>% rowSums()
term.freq <- term.freq %>% subset(term.freq >= 20)
df <- data.frame(term = names(term.freq), freq = term.freq)
```

```r
## plot frequent words
library(ggplot2)
ggplot(df, aes(x=term, y=freq)) + geom_bar(stat="identity") +
  xlab("Terms") + ylab("Count") + coord_flip() +
  theme(axis.text=element_text(size=7))
```



Wordcloud

```r
## word cloud
m <- tdm %>% as.matrix
# calculate the frequency of words and sort it by frequency
word.freq <- m %>% rowSums() %>% sort(decreasing = T)
# colors
library(RColorBrewer)
pal <- brewer.pal(9, "BuGn")[-(1:4)]
```

```r
# plot word cloud
library(wordcloud)
wordcloud(words = names(word.freq), freq = word.freq, min.freq = 3,
    random.order = F, colors = pal)
```

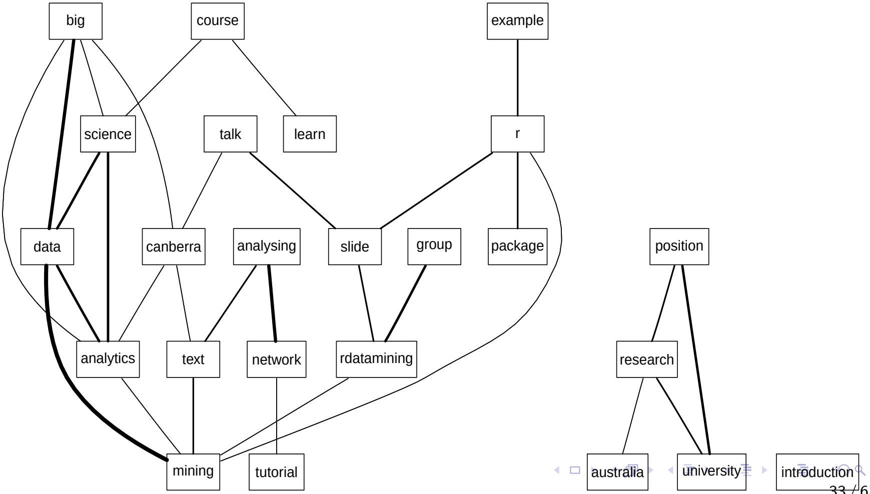

data
mining
r
slide
research
analysing
package
big
analytics
example
position
university
network
australia
tutorial
group
social
ausdm
course
science
learn
statistical
canberra
introduction
computational
rdatamining
series
program
document
conference
online
talk
present
useful
start
scientist
text
book
classification
predicting
postdoctoral
time
code
video
user
workshop
knowledge
august
application
available
machine
cluster
pdffree
lecture
dataset
modeling
submission
cfp
graph
poll
may
open
youtube
australian
performance
website
answers
support
member
southern
contain
share
california
improve
australasian
natural
san
engine
detailed
public
industrial
create
case
result
america
forecasting
thursday
technological
source
nov
outlier
jan
check
hadoop
follow
fast
large
make
sept
close
database
updated
visit
nice
spark
francisco
quick
call
business
provided
singapore
plot
link
card
pls
cloud
google
looking
track
lab
web
informal
cran
recent
will
build
mexico
reference
sna
us
top
center
project
extended
tricks
pm
map
oct
search
system
various
tweet
function
survey
software
go
nd
th
extract
guidance
advanced
detection
please
june
find
dr
v
paper
risk
list
developed
stanford
tool
add
iapa
download
forest
acm
join
event
short
tuesday
vacancies
feb
analyst
dynamic
technique
healthcare
edited
comment
fit
canada
twitter
handling
easier
world
dmapps
retrieval
thanks
decision
access
april
vs
now
file
can
prof
notes
postdoc
management
interacting
rstudio
distributed
get
new
little
massive
job
kdd
rule
excel
fellow
published
chapter
due
high
together
state
snowfall
webinar
official
week
step
friday
china
language
aug
give
page
mapreduce
skills
random
load
mid
tree
deadline
seminar
associate
sydney
process
sas
area
linkedin
kdnuggets
intern
seattle
participation
visualisations
graphical
credit
algorithm
sigkdd
run
melbourne
parallel
ieee
coursera
facebook
sunday
apache
senior
sentiment
march
keynote
iselect
version
today
titled
format
ranked
experience
neoj
initial
media
spatial
mode
regression
topic
amazon
studies
task
simple
wwwrdataminingcom
summit
competition
datacamp

Associations

```r
# which words are associated with 'r'?
tdm %>% findAssocs("r", 0.2)
## $r
##     code  example   series     user markdown
##     0.27     0.21     0.21     0.20     0.20

# which words are associated with 'data'?
tdm %>% findAssocs("data", 0.2)
## $data
##    mining       big analytics   science      poll
##      0.48      0.44      0.31      0.29      0.24
```

Network of Terms

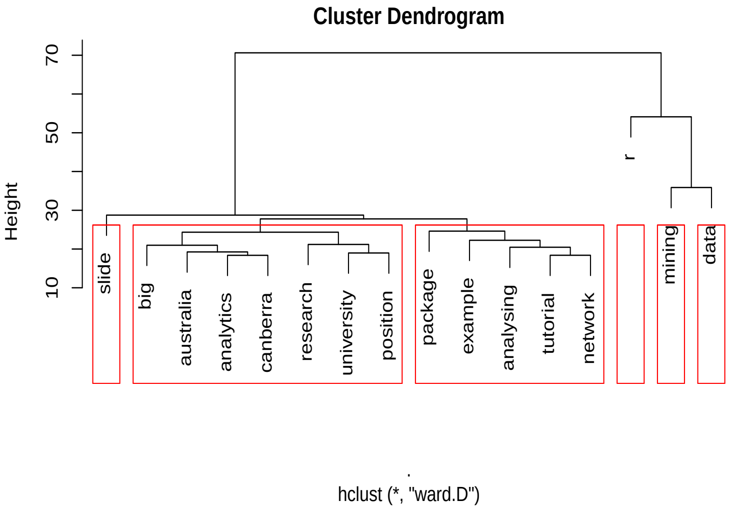
```r
## network of terms
library(graph)
library(Rgraphviz)
plot(tdm, term = freq.terms, corThreshold = 0.1, weighting = T)
```



Hierarchical Clustering of Terms

```r
## clustering of terms remove sparse terms
m2 <- tdm %>% removeSparseTerms(sparse = 0.95) %>% as.matrix()
# calculate distance matrix
dist.matrix <- m2 %>% scale() %>% dist()
# hierarchical clustering
fit <- dist.matrix %>% hclust(method = "ward")
```

```r
plot(fit)
fit %>% rect.hclust(k = 6)  # cut tree into 6 clusters
groups <- fit %>% cutree(k = 6)
```



```r
## k-means clustering of documents
m3 <- m2 %>% t()  # transpose the matrix to cluster documents (tweets)
set.seed(122)  # set a fixed random seed to make the result reproducibl
k <- 6  # number of clusters
kmeansResult <- kmeans(m3, k)
round(kmeansResult$centers, digits = 3)  # cluster centers
```

```
##   mining analytics australia  data canberra university slide
## 1  0.435     0.000     0.000 0.217    0.000      0.000 0.087
## 2  1.128     0.154     0.000 1.333    0.026      0.051 0.179
## 3  0.055     0.018     0.009 0.164    0.027      0.009 0.227
## 4  0.083     0.014     0.056 0.000    0.035      0.097 0.090
## 5  0.412     0.206     0.098 1.196    0.137      0.039 0.078
## 6  0.167     0.133     0.133 0.567    0.033      0.233 0.000
##   tutorial   big package     r network analysing research
## 1    0.043 0.000   0.043 1.130   0.087     0.174    0.000
## 2    0.026 0.077   0.282 1.103   0.000     0.051    0.000
## 3    0.064 0.018   0.109 1.127   0.045     0.109    0.000
## 4    0.056 0.007   0.090 0.000   0.090     0.111    0.000
## 5    0.059 0.333   0.010 0.020   0.020     0.059    0.020
## 6    0.000 0.167   0.033 0.000   0.067     0.100    1.233
##   position example
## 1    0.000   1.043
## 2    0.000   0.026
## 3    0.000   0.000
```

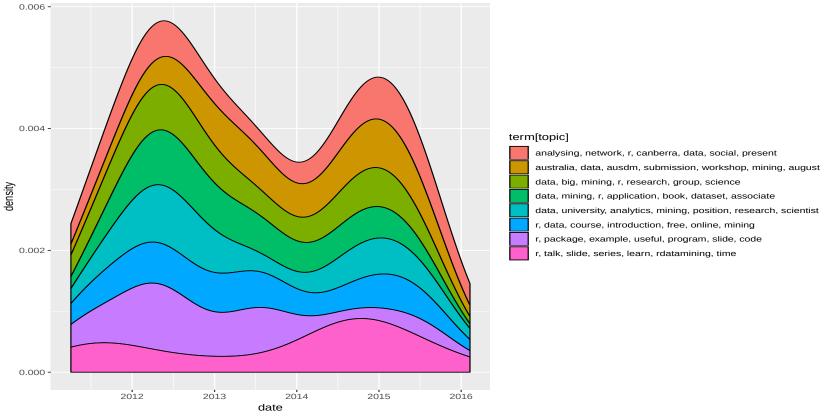
```r
for (i in 1:k) {
    cat(paste("cluster ", i, ":  ", sep = ""))
    s <- sort(kmeansResult$centers[i, ], decreasing = T)
    cat(names(s)[1:5], "\n")
    # print the tweets of every cluster
    # print(tweets[which(kmeansResult$cluster==i)])
}
## cluster 1:  r example mining data analysing
## cluster 2:  data mining r package slide
## cluster 3:  r slide data package analysing
## cluster 4:  analysing university slide package network
## cluster 5:  data mining big analytics canberra
## cluster 6:  research data position university mining
```

Topic Modelling

```
dtm <- tdm %>% as.DocumentTermMatrix()
library(topicmodels)
lda <- LDA(dtm, k = 8)  # find 8 topics
term <- terms(lda, 7)  # first 7 terms of every topic
term <- apply(term, MARGIN = 2, paste, collapse = ", ") %>% print
##                                                                 ...
##                     "data, big, mining, r, research, group,...
##                                                                 ...
##             "analysing, network, r, canberra, data, social,...
##                                                                 ...
##                   "r, talk, slide, series, learn, rdatamini...
##                                                                 ...
##                "r, data, course, introduction, free, online...
##                                                                 ...
##             "data, mining, r, application, book, dataset, a...
##                                                                 ...
##                  "r, package, example, useful, program, sli...
##                                                                 ...
## "data, university, analytics, mining, position, research, s...
##                                                                 ...
##       "australia, data, ausdm, submission, workshop, mining...
```

Topic Modelling

```
rdm.topics <- topics(lda) # 1st topic identified for every document (tw
rdm.topics <- data.frame(date=as.IDate(tweets.df$created),
                         topic=rdm.topics)
ggplot(rdm.topics, aes(date, fill = term[topic])) +
  geom_density(position = "stack")
```



Another way to plot steam graph:

http://menugget.blogspot.com.au/2013/12/data-mountains-and-streams-stacked-area.html

Sentiment Analysis

```
## sentiment analysis install package sentiment140
require(devtools)
install_github("sentiment140", "okugami79")
```

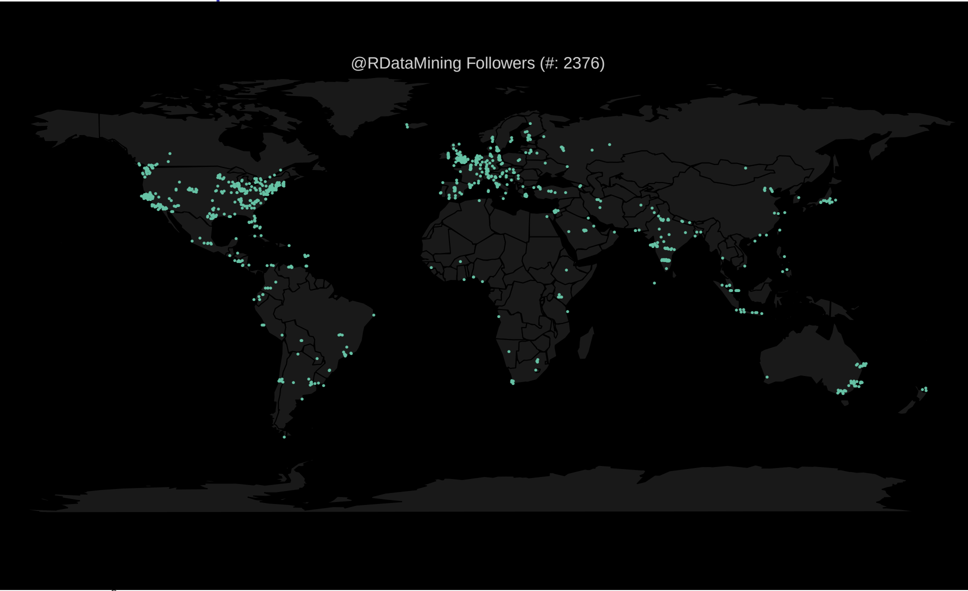
```
# sentiment analysis
library(sentiment)
sentiments <- sentiment(tweets.df$text)
table(sentiments$polarity)
# sentiment plot
sentiments$score <- 0
sentiments$score[sentiments$polarity == "positive"] <- 1
sentiments$score[sentiments$polarity == "negative"] <- -1
sentiments$date <- as.IDate(tweets.df$created)
result <- aggregate(score ~ date, data = sentiments, sum)
```

Retrieve User Info and Followers

```
## follower analysis
user <- getUser("RDataMining")
user$toDataFrame()
friends <- user$getFriends() # who this user follows
followers <- user$getFollowers() # this user's followers
followers2 <- followers[[1]]$getFollowers() # a follower's followers
```

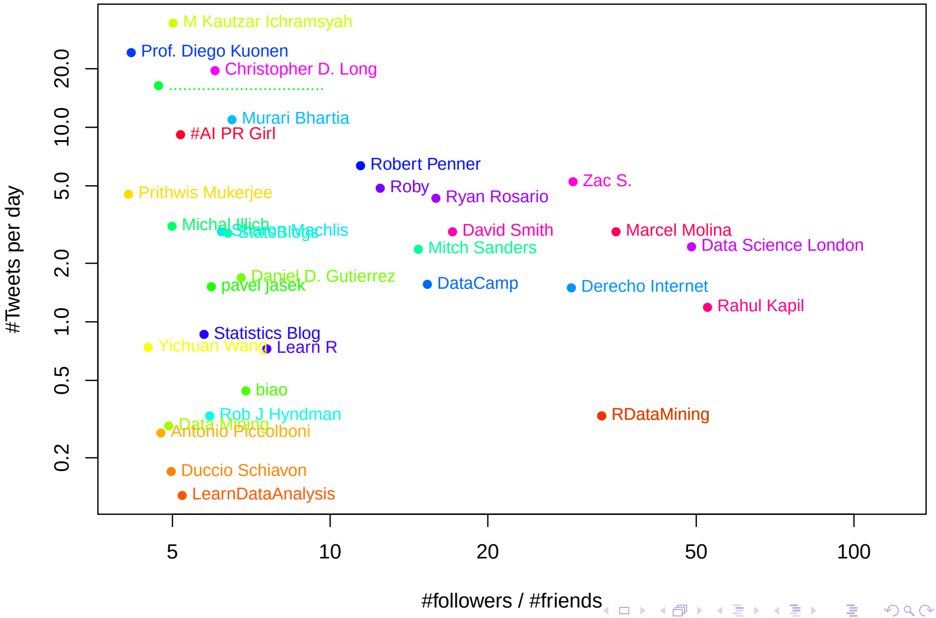
```
##                     [,1]                                         ...
## description         "R and Data Mining. Group on LinkedIn: ht...
## statusesCount       "583"                                        ...
## followersCount      "2376"                                       ...
## favoritesCount      "6"                                          ...
## friendsCount        "72"                                         ...
## url                 "http://t.co/LwL50uRmPd"                     ...
## name                "Yanchang Zhao"                              ...
## created             "2011-04-04 09:15:43"                        ...
## protected           "FALSE"                                      ...
## verified            "FALSE"                                      ...
## screenName          "RDataMining"                                ...
## location            "Australia"                                  ...
## lang                "en"                                         ...
```

Follower Map[§]



[§]Based on Jeff Leek's twitterMap function at
http://biostat.jhsph.edu/~jleek/code/twitterMap.R

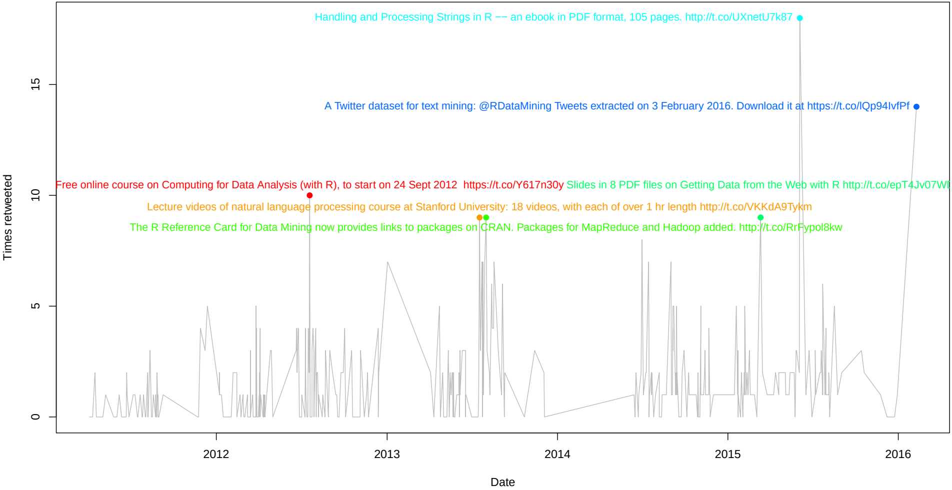
Active Influential Followers



Top Retweeted Tweets

```r
## retweet analysis
## select top retweeted tweets
table(tweets.df$retweetCount)
selected <- which(tweets.df$retweetCount >= 9)

## plot them
dates <- strptime(tweets.df$created, format="%Y-%m-%d")
plot(x=dates, y=tweets.df$retweetCount, type="l", col="grey",
     xlab="Date", ylab="Times retweeted")
colors <- rainbow(10)[1:length(selected)]
points(dates[selected], tweets.df$retweetCount[selected],
       pch=19, col=colors)
text(dates[selected], tweets.df$retweetCount[selected],
     tweets.df$text[selected], col=colors, cex=.9)
```

Top Retweeted Tweets



Tracking Message Propagation

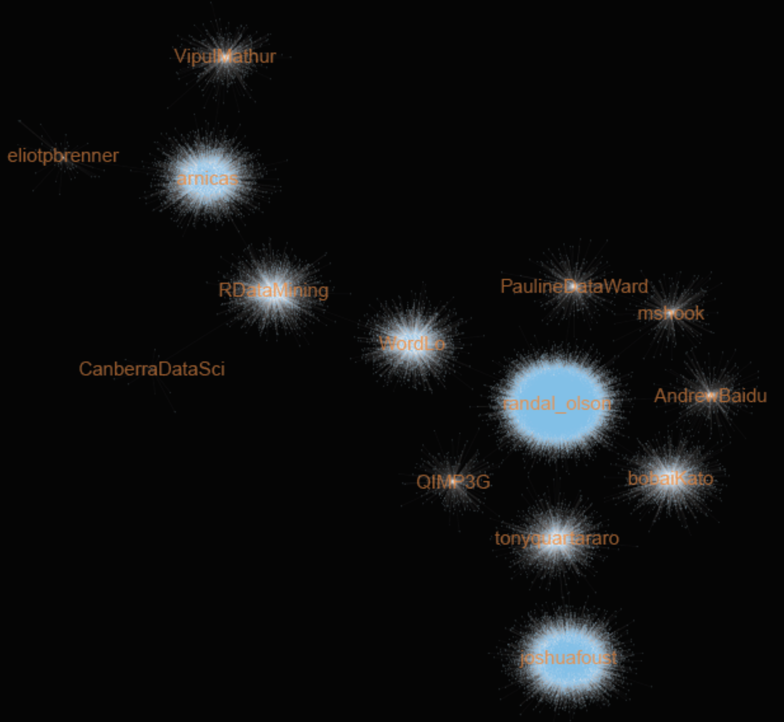
```
tweets[[1]]
retweeters(tweets[[1]]$id)
retweets(tweets[[1]]$id)
```

```
## [1] "RDataMining: A Twitter dataset for text mining: @RData...
```

```
##  [1] "197489286"  "316875164"  "229796464"  "3316009302"
##  [5] "244077734"  "16900353"   "2404767650" "222061895"
##  [9] "11686382"   "190569306"  "49413866"   "187048879"
## [13] "6146692"    "2591996912"
```

```
## [[1]]
## [1] "bobaiKato: RT @RDataMining: A Twitter dataset for text...
##
## [[2]]
## [1] "VipulMathur: RT @RDataMining: A Twitter dataset for te...
##
## [[3]]
## [1] "tau_phoenix: RT @RDataMining: A Twitter dataset for te...
```

The tweet potentially reached around 120,000 users.


VipulMathur
eliotpbrenner
arnicas
RDataMining
CanberraDataSci
PaulineDataWard
mshook
WordLo
randal_olson
AndrewBaidu
QIMP3G
bobaiKato
tonyquartararo
joshuafoust

Contents

R Packages

- Twitter data extraction: *twitteR*
- Text cleaning and mining: *tm*
- Word cloud: *wordcloud*
- Topic modelling: *topicmodels*, *lda*
- Sentiment analysis: *sentiment140*
- Social network analysis: *igraph*, *sna*
- Visualisation: *wordcloud*, *Rgraphviz*, *ggplot2*

Twitter Data Extraction – Package *twitteR* ¶

- `userTimeline`, `homeTimeline`, `mentions`, `retweetsOfMe`: retrive various timelines
- `getUser`, `lookupUsers`: get information of Twitter user(s)
- `getFollowers`, `getFollowerIDs`: retrieve followers (or their IDs)
- `getFriends`, `getFriendIDs`: return a list of Twitter users (or user IDs) that a user follows
- `retweets`, `retweeters`: return retweets or users who retweeted a tweet
- `searchTwitter`: issue a search of Twitter
- `getCurRateLimitInfo`: retrieve current rate limit information
- `twListToDF`: convert into data.frame

¶ https://cran.r-project.org/package=twitteR

Text Mining – Package *tm* [‖]

- `removeNumbers`, `removePunctuation`, `removeWords`, `removeSparseTerms`, `stripWhitespace`: remove numbers, punctuations, words or extra whitespaces
- `removeSparseTerms`: remove sparse terms from a term-document matrix
- `stopwords`: various kinds of stopwords
- `stemDocument`, `stemCompletion`: stem words and complete stems
- `TermDocumentMatrix`, `DocumentTermMatrix`: build a term-document matrix or a document-term matrix
- `termFreq`: generate a term frequency vector
- `findFreqTerms`, `findAssocs`: find frequent terms or associations of terms
- `weightBin`, `weightTf`, `weightTfIdf`, `weightSMART`, `WeightFunction`: various ways to weight a term-document matrix

[‖] https://cran.r-project.org/package=tm

Topic Modelling and Sentiment Analysis – Packages *topicmodels & sentiment140*

Package *topicmodels* [**]

- `LDA`: build a Latent Dirichlet Allocation (LDA) model
- `CTM`: build a Correlated Topic Model (CTM) model
- `terms`: extract the most likely terms for each topic
- `topics`: extract the most likely topics for each document

Package *sentiment140* [††]

- `sentiment`: sentiment analysis with the sentiment140 API, tune to Twitter text analysis

[**] https://cran.r-project.org/package=topicmodels
[††] https://github.com/okugami79/sentiment140

Social Network Analysis and Visualization – Package *igraph* [‡‡]

- `degree`, `betweenness`, `closeness`, `transitivity`: various centrality scores
- `neighborhood`: neighborhood of graph vertices
- `cliques`, `largest.cliques`, `maximal.cliques`, `clique.number`: find cliques, ie. complete subgraphs
- `clusters`, `no.clusters`: maximal connected components of a graph and the number of them
- `fastgreedy.community`, `spinglass.community`: community detection
- `cohesive.blocks`: calculate cohesive blocks
- `induced.subgraph`: create a subgraph of a graph (*igraph*)
- `read.graph`, `write.graph`: read and writ graphs from and to files of various formats

[‡‡] https://cran.r-project.org/package=igraph

Contents

Wrap Up

- Transform unstructured data into structured data (i.e., term-document matrix), and then apply traditional data mining algorithms like clustering and classification
- Feature extraction: term frequency, TF-IDF and many others
- Text cleaning: lower case, removing numbers, puntuations and URLs, stop words, stemming and stem completion
- Stem completion may not always work as expected.
- Documents in languages other than English

Contents

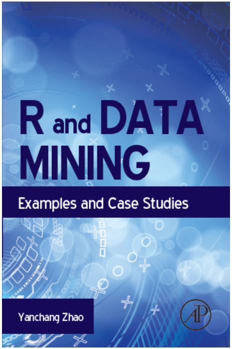
Further Readings

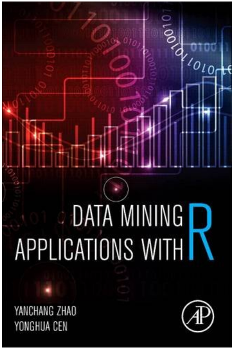
- Text Mining
  https://en.wikipedia.org/wiki/Text_mining
- TF-IDF
  https://en.wikipedia.org/wiki/Tf\OT1\textendashidf
- Topic Modelling
  https://en.wikipedia.org/wiki/Topic_model
- Sentiment Analysis
  https://en.wikipedia.org/wiki/Sentiment_analysis
- Document Summarization
  https://en.wikipedia.org/wiki/Automatic_summarization
- Natural Language Processing
  https://en.wikipedia.org/wiki/Natural_language_processing
- An introduction to text mining by Ian Witten
  http://www.cs.waikato.ac.nz/%7Eihw/papers/04-IHW-Textmining.pdf

Online Resources

- Book titled *R and Data Mining: Examples and Case Studies* [Zhao, 2012]
  http://www.rdatamining.com/docs/RDataMining-book.pdf
- R Reference Card for Data Mining
  http://www.rdatamining.com/docs/RDataMining-reference-card.pdf
- Free online courses and documents
  http://www.rdatamining.com/resources/
- RDataMining Group on LinkedIn (27,000+ members)
  http://group.rdatamining.com
- Twitter (3,300+ followers)
  @RDataMining

The End





Thanks!

Email: `yanchang(at)RDataMining.com`
Twitter: `@RDataMining`

How to Cite This Work

- Citation
  Yanchang Zhao. R and Data Mining: Examples and Case Studies. ISBN 978-0-12-396963-7, December 2012. Academic Press, Elsevier. 256 pages. URL: http://www.rdatamining.com/docs/RDataMining-book.pdf.
- BibTex

```
@BOOK{Zhao2012R,
    title = {R and Data Mining: Examples and Case Studies},
    publisher = {Academic Press, Elsevier},
    year = {2012},
    author = {Yanchang Zhao},
    pages = {256},
    month = {December},
    isbn = {978-0-123-96963-7},
    keywords = {R, data mining},
    url = {http://www.rdatamining.com/docs/RDataMining-book.pdf}
}
```

References I

Zhao, Y. (2012).
R and Data Mining: Examples and Case Studies, ISBN 978-0-12-396963-7.
Academic Press, Elsevier.

Zhao, Y. (2013).
Analysing twitter data with text mining and social network analysis.
In *Proc. of the 11th Australasian Data Mining Conference (AusDM 2013)*, Canberra, Australia.