Advanced Analytics with R

Text Analytics

Text Analytics

- ***Text analytics***, as known as ***text mining***, is the process of obtaining patterns and trends from text.
- Typical text analytics tasks include
  - text categorization
  - text clustering
  - concept/entity extraction
  - production of granular taxonomies
  - *sentiment analysis*
  - document summarization, and
  - entity relation modeling

Applications of Text Analytics

Applications of Text Analytics

Applications of Text Analytics

Getting Twitter Data

- https://sites.google.com/site/miningtwitter/basics/getting-data/by-twitter
- https://www.credera.com/blog/business-intelligence/twitter-analytics-using-r-part-1-extract-tweets/

Getting Facebook Data

- https://bigdataenthusiast.wordpress.com/2016/03/19/mining-facebook-data-using-r-facebook-api/

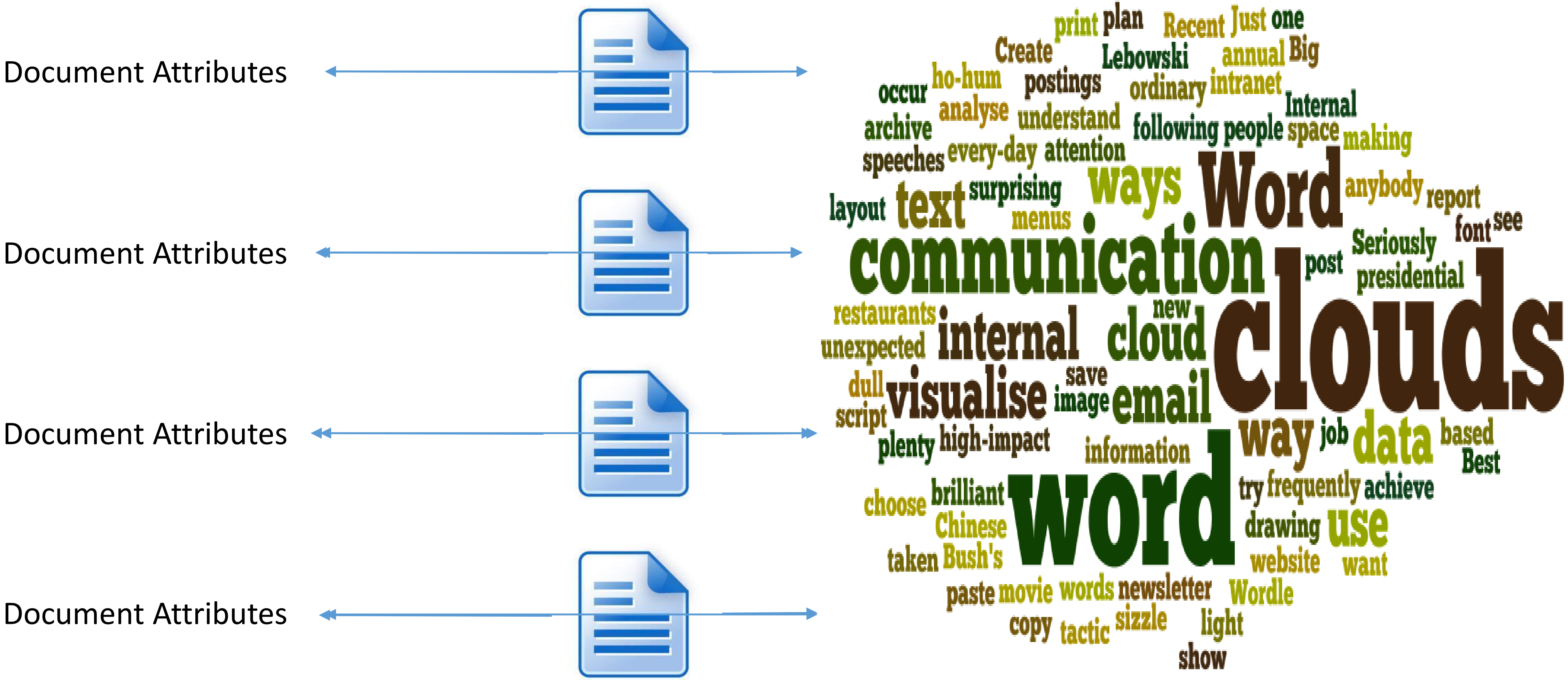
Getting Instagram Data

- https://www.r-bloggers.com/analyze-instagram-with-r/

Case – Movie Review

- Data: Reviews on 1000 movies from the Internet Movie Database (IMDb).
- Task: to study whether we can predict the movie rating or sentiment (thumbs up or down) from the reviews.

Basic Concept



Document Attributes

Document Attributes

Document Attributes

Document Attributes

print plan Recent Just one
Create Lebowski annual Big
ho-hum postings ordinary intranet
occur analyse understand Internal
archive following people space making
speeches every-day attention
layout text surprising ways Word anybody report
menus font see
communication Seriously
post presidential
restaurants new
unexpected internal cloud clouds
dull visualise save image email
script way job data based
plenty high-impact information Best
choose brilliant word try frequently achieve
Chinese drawing use
taken Bush's website want
paste movie words newsletter Wordle
copy tactic sizzle light
show

Step 1. Data Preparation

Corpus

- Normally, texts are originally stored in various documents, each of which has its own attributes, such as date, owner, rating, etc.
- A collection of text documents are stored in a structure called ***Corpus*** for exploration.
- The default implementation is the so-called ***VCorpus*** (short for Volatile Corpus) which realizes a semantics as known from most R objects: corpora are R objects held fully in memory. We denote this as volatile since once the R object is destroyed, the whole corpus is gone.
- Another implementation is the ***PCorpus*** which implements a Permanent Corpus semantics, i.e., the documents are physically stored outside of R (e.g., in a database), corresponding R objects are basically only pointers to external structures, and changes to the underlying corpus are reflected to all R objects associated with it. Compared to the volatile corpus the corpus encapsulated by a permanent corpus object is not destroyed if the corresponding R object is released

Read in data

```
#Let's first load data stored in /reviews/train/unsup/, which are the unsupervised data
directory.location <- paste(getwd(),"/reviews/train/unsup/",sep = "")

#Read in data and stored it as a Corpus
movie.unsup.corpus <- Corpus(DirSource(directory.location, encoding = "UTF-8"),
  readerControl = list(language = "en_US"))
```

Inspect Corpus

```
#check the number of documents stored in a corpus
length(movie.unsup.corpus)

#Have an overview of (part of) the corpus
inspect(movie.unsup.corpus[1])

#view content of a particular document in a corpus
movie.unsup.corpus[[1]]$content

#You can even use lapply function on corpus
review.list <- lapply(movie.unsup.corpus, as.character)
```

Data Transformation

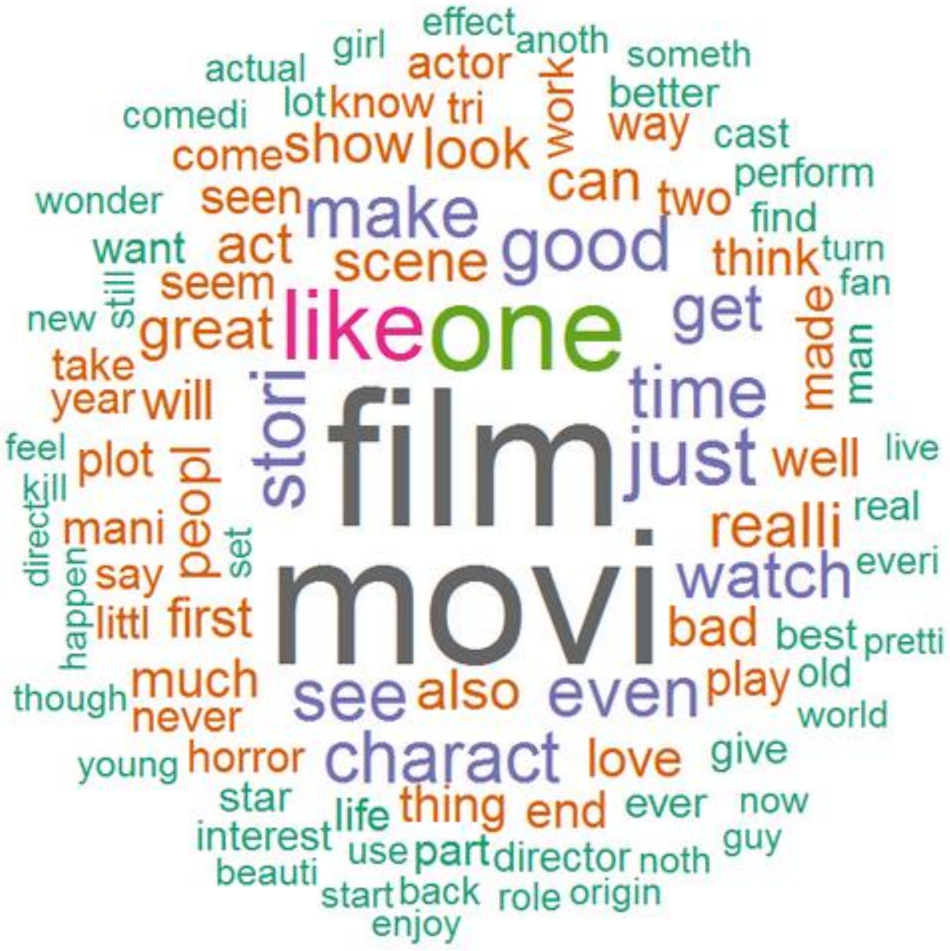
- Steps are to be perform to transform documents to make the subsequent analysis more meaningful and more efficient:
  - Make the words more consistent
    - Good = good
    - Wants = want = wanted = wanting
  - Remove unnecessary words
    - White space
    - Punctuations
    - Numbers
    - Stop words

Further data transformation

- Identify top words in the corpus
- Inspect all the top words so that we can
    - Remove meaningless words
    - Correct key words

Word cloud

```
wordcloud(movie.unsup.corpus,max.words=100,random.order=FALSE,colors=brewer.pal(8, "Dark2"))
```



Sentiment Analysis

Load Positive and Negative Dictionary

```
bytecode.convert <- function(x) {iconv(enc2utf8(x), sub = "byte")}

# read in positive and negative word lists from Hu and Liu (2004)
positive.data.frame <- read.table(file = "Hu_Liu_positive_word_list.txt",
  header = FALSE, colClasses = c("character"), row.names = NULL,
  col.names = "positive.words")
positive.data.frame$positive.words <-
  bytecode.convert(positive.data.frame$positive.words)

negative.data.frame <- read.table(file = "Hu_Liu_negative_word_list.txt",
  header = FALSE, colClasses = c("character"), row.names = NULL,
  col.names = "negative.words")
negative.data.frame$negative.words <-
  bytecode.convert(negative.data.frame$negative.words)
```

Detecting Top Positive Words

```
Hu.Liu.positive.dictionary <-
    c(as.character(positive.data.frame$positive.words))
reviews.tdm.Hu.Liu.positive <- TermDocumentMatrix(movie.unsup.corpus,
  list(dictionary = Hu.Liu.positive.dictionary))
examine.tdm <- removeSparseTerms(reviews.tdm.Hu.Liu.positive, 0.95)
top.words <- Terms(examine.tdm)
print(top.words)
Hu.Liu.frequent.positive <- findFreqTerms(reviews.tdm.Hu.Liu.positive, 25)
# this provides a list positive words occurring at least 25 times
# a review of this list suggests that all make sense (have content validity)
# Then we accept Hu.Liu.frequent.positive fully as our positive dictionary
test.positive.dictionary <- c(as.character(Hu.Liu.frequent.positive))
```

Detecting Top Negative Words

```
Hu.Liu.negative.dictionary <-
    c(as.character(negative.data.frame$negative.words))
reviews.tdm.Hu.Liu.negative <- TermDocumentMatrix(movie.unsup.corpus,
  list(dictionary = Hu.Liu.negative.dictionary))
examine.tdm <- removeSparseTerms(reviews.tdm.Hu.Liu.negative, 0.97)
top.words <- Terms(examine.tdm)
print(top.words)
Hu.Liu.frequent.negative <- findFreqTerms(reviews.tdm.Hu.Liu.negative, 15)

# this provides a short list negative words occurring at least 15 times
# across the document collection...
# By checking the list, we found that one of these words seems out of place
# as they could be thought of as positive: "funny"
test.negative <- setdiff(Hu.Liu.frequent.negative,c("funny"))
# Now we exclude "funny" from the negative dictionary
test.negative.dictionary <- c(as.character(test.negative))
```

From Text to Numbers

	Doc1	Doc2	…	Doc1000
Good	1	0		2
Amaze	0	1		0
⋮				
Favorite	1	0		1
Fall	1	0		1
Dead	2	1		0
⋮				
Hell	0	0		3

TermDocumentMatrix

	Doc1	Doc2	…	Doc1000
#Total words	230	340		135
#Positive words	20	30		2
#Negative words	10	20		13
#Other words	200	290		120
Rating	10	5		2

Predictive Analysis

Simple classification

- Classify a movie into thumbs up (rating >5) or thumbs down (rating <=5) by checking if there is more positive words or more negative words in the movie review.

Linear Regression

- Predict rating by linear regression on various numbers of words.
- Predicting thumbs up or down by using the linear regression result.

Further Study

- Applying other classification methods
- Weighted regression on various key words
- Cluster analysis