

DSI Summer Workshops Series

June 14, 2018

Peggy Lindner

Center for Advanced Computing & Data Science (CACDS)

Data Science Institute (DSI)

University of Houston

plindner@uh.edu

Please make sure you have a copy of R up and running, as well as a Python 3 installation (ideally from Anaconda).

Goals for today

Understand basics of text analysis using R

(well enough so that you can Google your problems, find the answer, and implement it.)

More specifically

1. Up and running with R & IPython
2. Understand a basic exploratory data analysis workflow
3. Basics of R and Topic Modeling

Why R and not Python

It's good for data exploration!

Part 1: Getting yourself ready

First: Install software on your computer

- R [CRAN \(https://www.anaconda.com/download/\)](https://www.anaconda.com/download/)
- Python [Anaconda \(https://www.anaconda.com/download/\)](https://www.anaconda.com/download/)

Second: Prep your R environment

On a Mac open a terminal and start R

```
[plindner@peggys-mbp:~$ R

R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

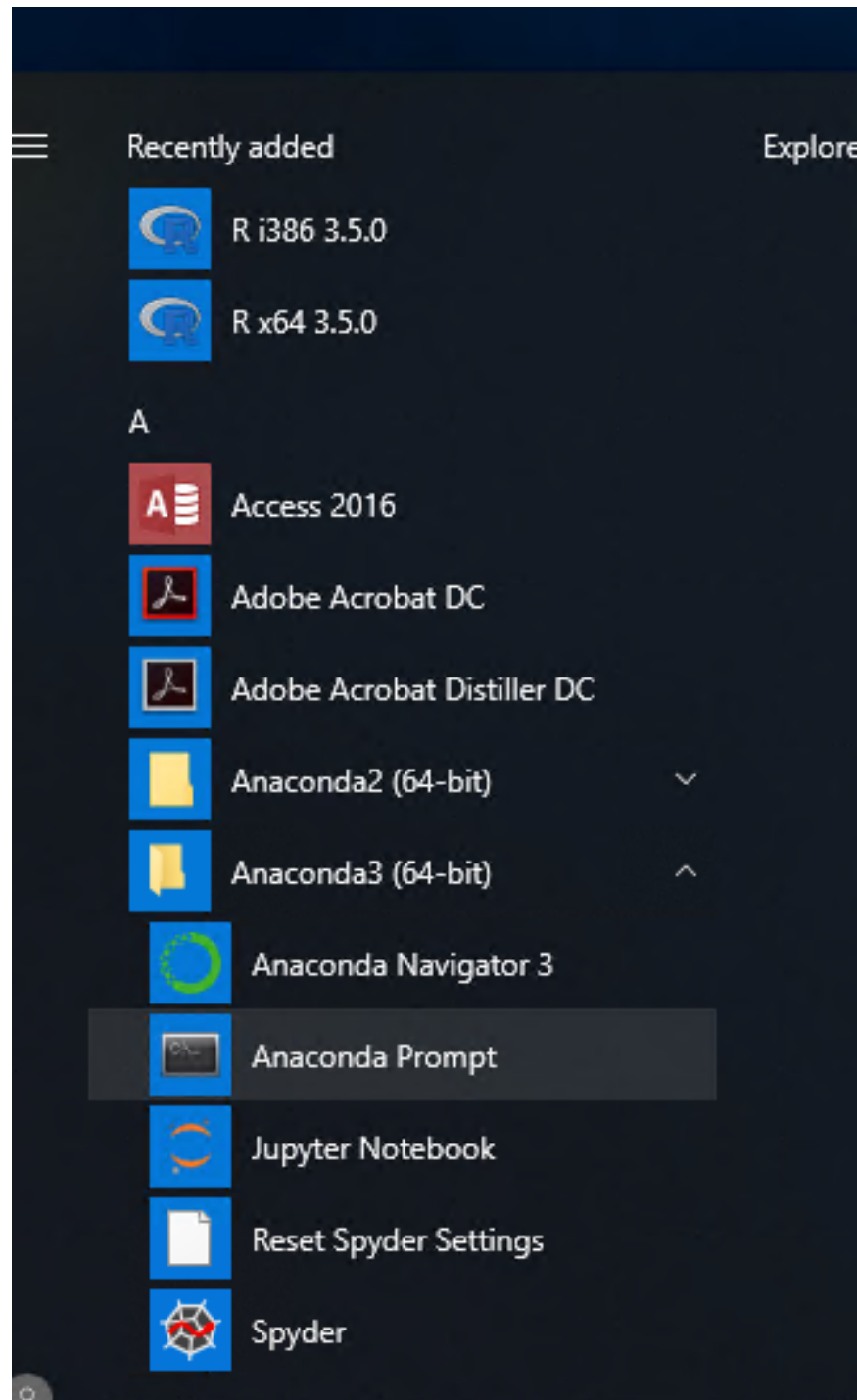
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> █
```

On Windows: Open the Anaconda Command line and start R



```
(C:\ProgramData\Anaconda3) C:\> cd C:\Program Files\R\R-3.5.0\bin\x64\
(C:\ProgramData\Anaconda3) C:\Program Files\R\R-3.5.0\bin\x64>R

R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

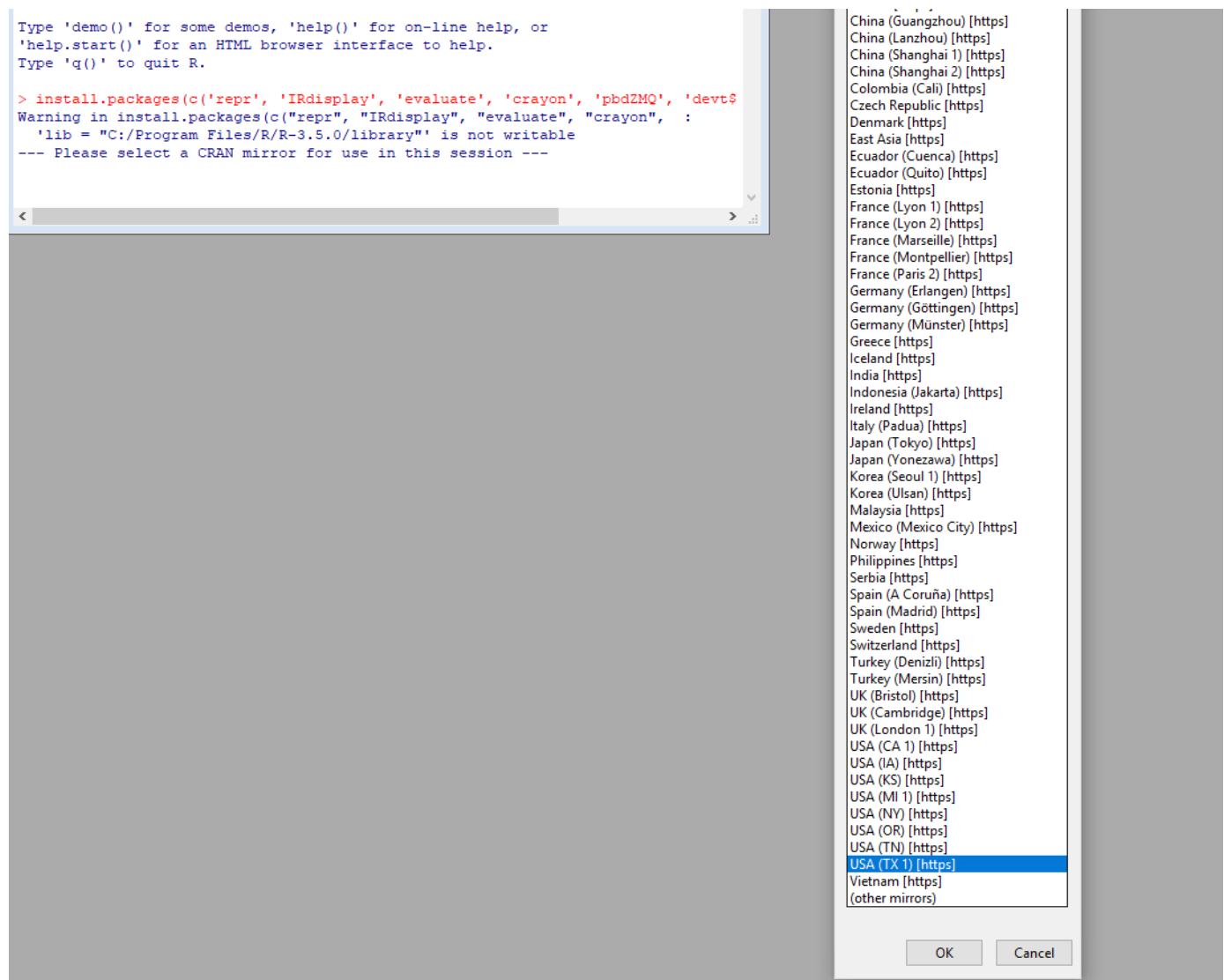
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> IRkernel::installspec()
[InstallKernelSpec] Installed kernelspec ir in C:\Users\plindner\AppData\Roaming\jupyter\kernels\ir
> q()
```

Now let's install some packages ...

```
> install.packages(c('readr', 'stringr', 'SnowballC', 'wordcloud', 'RColorBrewer'))
> install.packages(c('tm', 'ggplot2', 'topicmodels'))
> install.packages(c('repr', 'IRdisplay', 'evaluate', 'crayon', 'pbdZMQ', 'devtools', 'uuid', 'digest'))
> devtools::install_github('IRkernel/IRkernel')
```

When you see "Please select a CRAN mirror" , well select one.



... one last step - installing the Kernel

```
> IRkernel::installspec()
```

Now we can close the R environment (but leave your terminal and console open)

```
> quit()
```

Say "N" (no) when asked to save the workspace.

Jupyter Notebooks is what we will be going to use

We are now ready to start up our Jupyter Environment from the terminal or the console:

```
$ jupyter notebook --notebook-dir C:/Users/[your username]  
e]
```

or on a Mac

```
$ jupyter notebook --notebook-dir /Users/[your username]
```

And your browser should open at the address: <http://localhost:8888/tree>
(<http://localhost:8888/tree>)



Open the downloaded notebook on your computer

 Quit Logout

Files Running Clusters

Select items to perform actions on them. Upload New ▾ 

☐ 0 ▾

/ Downloads

NameLast Modified ↑File size

<input type="checkbox"/>	..	seconds ago	
<input type="checkbox"/>	 DSI Workshop Summer 2018a.ipynb	11 minutes ago	6.99 kB
<input type="checkbox"/>	 Simple_0.6.tar.gz	7 hours ago	223 kB
<input type="checkbox"/>	 Simple	7 hours ago	

Quick intro to Jupyter notebooks

Cells can be Markdown (like this one) or code

To start off with

Make sure you hit `Shift-Enter` or `Ctrl-Enter` when you are done. You can also use the "Run" button.

In []:

```
2 + 2
```

Part 2: The Exploratory Analysis Workflow

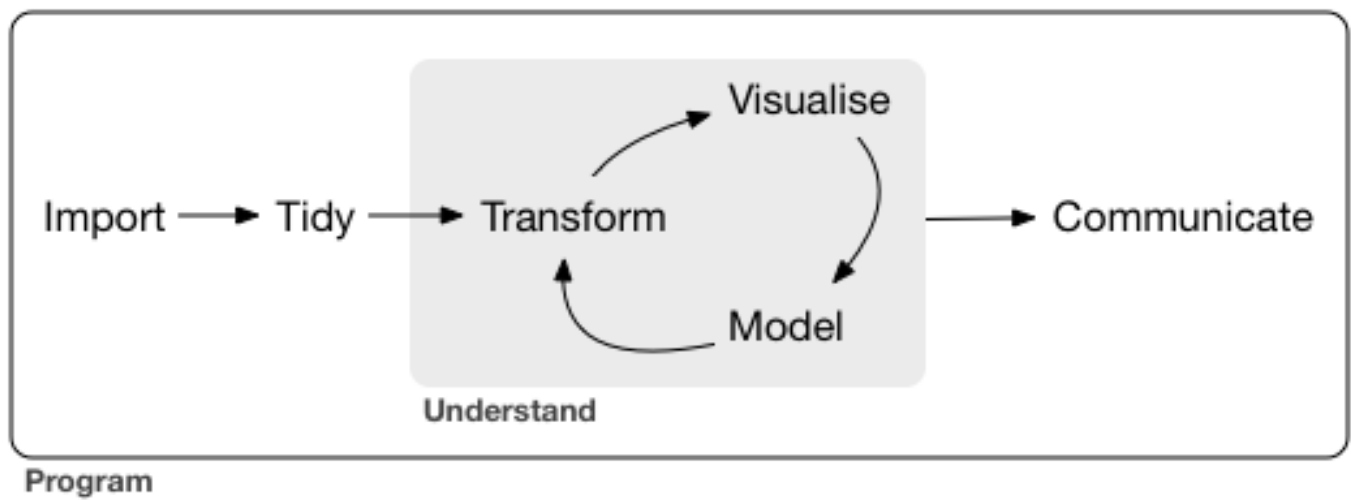
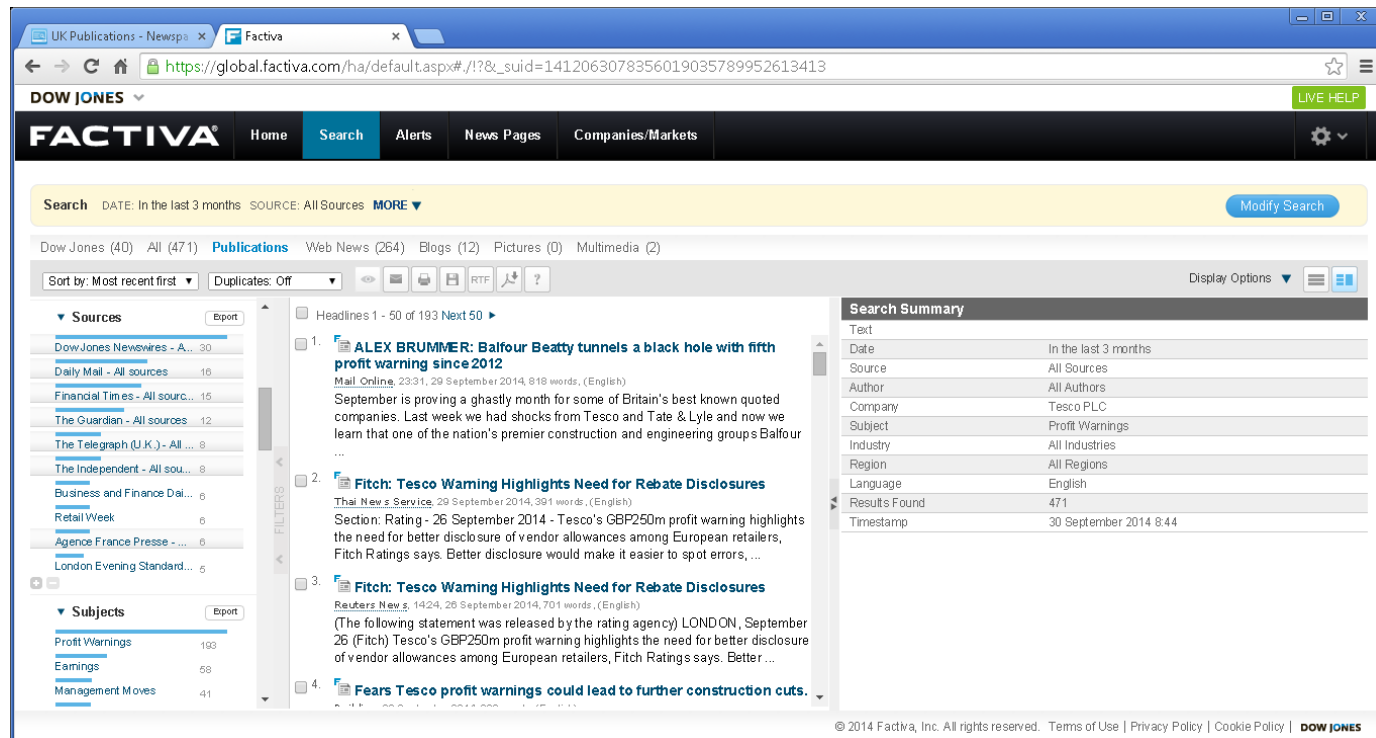


Image source: Hadley Wickham, R for Data Science

Our Example

Media Analysis of a bunch of articles downloaded from a database called "Factiva"



The screenshot shows the Factiva search results page for Dow Jones publications. The page is titled "DOW JONES" and "FACTIVA". The search criteria are "DATE: In the last 3 months" and "SOURCE: All Sources". The results are sorted by "Most recent first" and show 471 results. The first result is "ALEX BRUMMER: Balfour Beatty tunnels a black hole with fifth profit warning since 2012". The second result is "Fitch: Tesco Warning Highlights Need for Rebate Disclosures". The third result is "Fitch: Tesco Warning Highlights Need for Rebate Disclosures". The fourth result is "Fears Tesco profit warnings could lead to further construction cuts".

Search Summary

Text	
Date	In the last 3 months
Source	All Sources
Author	All Authors
Company	Tesco PLC
Subject	Profit Warnings
Industry	All Industries
Region	All Regions
Language	English
Results Found	471
Timestamp	30 September 2014 8:44

Make sure you download the data source file: <http://bit.ly/UHDSldata1> (<http://bit.ly/UHDSldata1>) and store it in the Jupyter notebook directory (next to the *.ipynb file)

Frequently used R Packages in conjunction with text data

- [readr](https://cran.r-project.org/web/packages/readr/readr.pdf) (<https://cran.r-project.org/web/packages/readr/readr.pdf>) Import data

Data Analysis of text based material

- [stringr](https://cran.r-project.org/web/packages/stringr/vignettes/stringr.html) (<https://cran.r-project.org/web/packages/stringr/vignettes/stringr.html>) Clean up text
- [SnowballC](https://cran.r-project.org/web/packages/SnowballC/SnowballC.pdf) (<https://cran.r-project.org/web/packages/SnowballC/SnowballC.pdf>) Stemming of words
- [tm](https://cran.r-project.org/web/packages/tm/vignettes/tm.pdf) (<https://cran.r-project.org/web/packages/tm/vignettes/tm.pdf>) Text mining
- [Quanteda](https://quanteda.io/) (<https://quanteda.io/>) versatile text analysis tool
- [topicmodels](https://www.tidytextmining.com/topicmodeling.html) (<https://www.tidytextmining.com/topicmodeling.html>) Topic Modeling

Visualization

- [ggplot2](http://ggplot2.tidyverse.org/) (<http://ggplot2.tidyverse.org/>) Modern R visualizations
- [wordcloud](http://developer.marvel.com) (<http://developer.marvel.com>) Make some nice word clouds
- [RColorBrewer](https://dataset.readthedocs.org/en/latest/) (<https://dataset.readthedocs.org/en/latest/>) Get color into your visualizations

In []:

```
#load all required libraries  
library(readr)  
library(stringr)  
library(SnowballC)  
library(wordcloud)  
library(RColorBrewer)  
library(tm)  
library(ggplot2)  
library(topicmodels)
```

Data Import

In []:

```
# put the name of your csv file
inputfile <- "AJA_Factiva.txt"
# read the data
alldata <- read_file(inputfile)
# look at the data
# what type is our data?
str(alldata)
```

Prepare data

In []:

```
# data wrangling - split the file in different articles
split.word <- "Document AJAZEN(.*)"

# split up into individual documents
list_alldataSplitted <- str_split(alldata, split.word)
# convert to vector and remove last element (which is a leftover)
alldataSplitted <- unlist(list_alldataSplitted)
alldataSplitted <- alldataSplitted[-length(alldataSplitted)]
str(alldataSplitted)
```

In []:

```
### create corpus
article.corpus <- Corpus(VectorSource(alldataSplitted))

article.corpus
```

In []:

```
#inspect a particular document
writeLines(as.character(article.corpus[[30]]))
```

In []:

```
#Check details (look at bunched up corpus to find anomalies)  
inspect(article.corpus)
```

Data cleaning

In []:

```
#create the toSpace content transformer  
toSpace <- content_transformer(function(x, pattern) { return (  
  gsub(pattern, " ", x)})}  
#to remove potentially problematic symbols  
article.corpus <- tm_map(article.corpus, toSpace, "-")  
article.corpus <- tm_map(article.corpus, toSpace, ":")  
article.corpus <- tm_map(article.corpus, toSpace, "'")  
article.corpus <- tm_map(article.corpus, toSpace, "'")  
article.corpus <- tm_map(article.corpus, toSpace, " -")  
  
#Good practice to check after each step.  
writeLines(as.character(article.corpus[[30]]))
```

In []:

```
#Remove punctuation - replace punctuation marks with " "  
article.corpus <- tm_map(article.corpus, removePunctuation)  
  
#Good practice to check after each step.  
writeLines(as.character(article.corpus[[30]]))
```

In []:

```
#Transform to lower case
article.corpus <- tm_map(article.corpus,content_transformer(to
lower))

#Strip digits
article.corpus <- tm_map(article.corpus, removeNumbers)

#Remove stopwords from standard stopword list
article.corpus <- tm_map(article.corpus, removeWords, stopword
s("english"))

#inspect output
writeLines(as.character(article.corpus[[30]]))
```

Stopwords (<https://github.com/arc12/Text-Mining-Weak-Signals/wiki/Standard-set-of-english-stopwords>)

In []:

```
#define and eliminate all custom stopwords
myStopwords <- c("monday")
article.corpus <- tm_map(article.corpus, removeWords, myStopwo
rds)

#Strip whitespace (cosmetic?)
article.corpus <- tm_map(article.corpus, stripWhitespace)

#inspect output
writeLines(as.character(article.corpus[[30]]))
```

Word Stemming (<http://www.omegahat.net/Rstem/stemming.pdf>)

In []:

```
#Stem document
article.corpus <- tm_map(article.corpus,stemDocument)

#inspect output
writeLines(as.character(article.corpus[[30]]))
```

Prepare for Analysis - create word counts

In []:

```
#Create document-term matrix
dtm <- DocumentTermMatrix(article.corpus)

dtm
```

In []:

```
#inspect segment of document term matrix
inspect(dtm[15:16,100:105])
```

In []:

```
#collapse matrix by summing over columns - this gets total counts (over all docs) for each term
freq <- colSums(as.matrix(dtm))
#length should be total number of terms
length(freq)
```

In []:

```
#create sort order (descending)
ord <- order(freq,decreasing=TRUE)
#inspect most frequently occurring terms
freq[head(ord)]
#inspect least frequently occurring terms
freq[tail(ord)]

#List all terms in decreasing order of freq and write to disk
write.csv(freq[ord],"word_freq.csv")
```

In []:

```
#alterantive: remove very frequent and very rare words
dtmr <- DocumentTermMatrix(article.corpus, control=list(wordLengths=c(4, 20),
                                                    bounds = list(global = c(3,2
7))))

dtmr

freqr <- colSums(as.matrix(dtmr))
#length should be total number of terms
length(freqr)

#create sort order (desc)
ordr <- order(freqr,decreasing=TRUE)
#inspect most frequently occurring terms
freqr[head(ordr)]
#inspect least frequently occurring terms
freqr[tail(ordr)]
```

In []:

```
#list most frequent terms. Lower bound specified as second argument
findFreqTerms(dtmr,lowfreq=60)
```

Now that we have the most frequently occurring terms in hand, we can check for correlations between some of these and other terms that occur in the corpus. In this context, correlation is a quantitative measure of the co-occurrence of words in multiple documents.

In []:

```
#correlations
findAssocs(dtm,"turkish",0.5)
findAssocs(dtm,"children",0.5)
```

One needs to specify the DTM, the term of interest and the correlation limit. The latter is a number between 0 and 1 that serves as a lower bound for the strength of correlation between the search and result terms. For example, if the correlation limit is 1, findAssocs() will return only those words that always co-occur with the search term. A correlation limit of 0.5 will return terms that have a search term co-occurrence of at least 50% and so on.

Visualizations

In []:

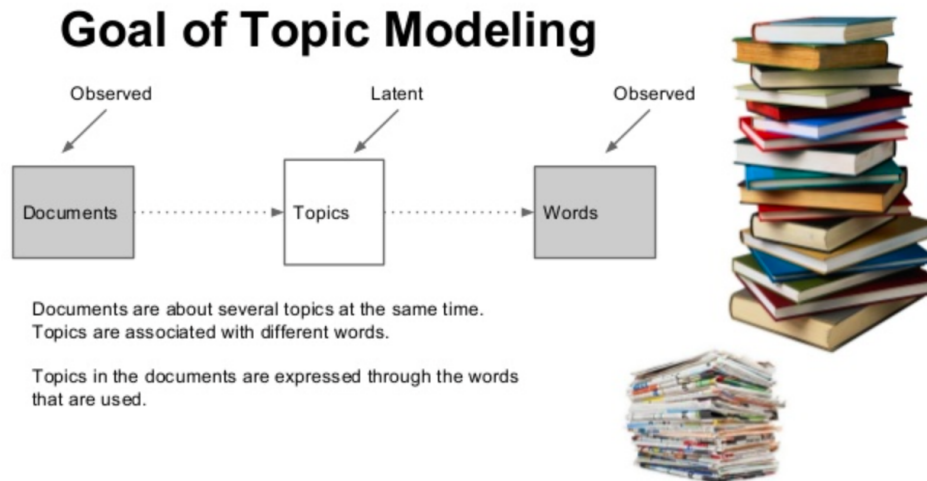
```
#Basic graphics  
#histogram  
wf=data.frame(term=names(freq),occurrences=freq)  
  
p <- ggplot(subset(wf, freq>50), aes(term, occurrences))  
p <- p + geom_bar(stat="identity")  
p <- p + theme(axis.text.x=element_text(angle=45, hjust=1))  
p
```

In []:

```
#wordcloud  
#setting the same seed each time ensures consistent look across clouds  
set.seed(42)  
#limit words by specifying min frequency  
wordcloud(names(freq),freq, min.freq=70)  
#...add color  
wordcloud(names(freq),freq,min.freq=70,colors=brewer.pal(6,"Dark2"))
```

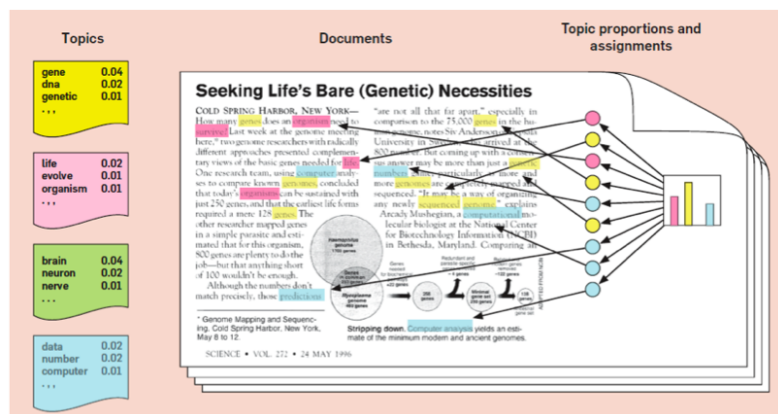

What is Topic Modeling

- deals with the problem of automatically classifying sets of documents into themes



What is behind Topic Modeling?

- Latent Dirichlet Allocation (LDA) ...
- ... assumes that each of the documents in a collection consist of a mixture of collection-wide topics
- in reality we observe only documents and words, not topics – the latter are part of the hidden (or latent) structure of documents
- goal is to infer the latent topic structure given the words and document - LDA does this by recreating the documents in the corpus by adjusting the relative importance of topics in documents and words in topics iteratively



In []:

```
#Topic modeling
#Set parameters for Gibbs sampling
burnin <- 4000
iter <- 2000
thin <- 500
seed <-list(2003,5,63,100001,765)
nstart <- 5
best <- TRUE

#Number of topics
k <- 5

#Run LDA using Gibbs sampling
ldaOut <-LDA(dtm,k, method="Gibbs", control=list(nstart=nstart
, seed = seed, best=best, burnin = burnin, iter = iter, thin=t
hin))
```

In []:

```
#have a look at the model and some output
ldaOut
topics(ldaOut)
as.matrix(terms(ldaOut,6))

#write out results
#docs to topics
ldaOut.topics <- as.matrix(topics(ldaOut))
write.csv(ldaOut.topics,file=paste("LDAGibbs",k,"DocsToTopics.
csv"))

#top 6 terms in each topic
ldaOut.terms <- as.matrix(terms(ldaOut,6))
write.csv(ldaOut.terms,file=paste("LDAGibbs",k,"TopicsToTerms.
csv"))
```