Data Science Project

Gege Gui

9/11/2017

Introduction:

PLoS (Public Library of Science) is a nonprofit Open Access publisher, innovator and advocacy organization with a mission to accelerate progress in science and medicine by leading a transformation in research communication. The data analysis project is to find out the most common statistical techniques in all published PLoS papers and the trends over the last 10-15 years.

Data:

Follow the documentation from PLoS API website http://api.plos.org/solr/examples/ to download statistics-related paper. By indicating "statistic" in abstract part, the results show that there are 12329 papers.

bulk downloading ftp://ftp.ncbi.nlm.nih.gov/pub/pmc/

Specify the "limit" to be 12329, then the data part for "id" contains the ID and publication date of the paper of our interest.

To use regular expression for further search, we create a dictionary for common statistical methods. We use "The Elements of Statistical Learning - Data Mining, Inference, and Prediction" second edition, by Trevor Hastie, Robert Tibshirani, Jerome Friedman.

```r
# create dictionary and return abstract
plosabstract(q = 'regression', fl='id,title', limit = 5)
plosabstract(q = 'machine learning', fl='id,title', limit = 5)
plosabstract(q = 'neural network', fl='id,title', limit = 5)
```

```r
# select paper with key words in method part
out <- highplos(q='regression', hl.fl = 'Materials and Methods', rows=130)
# highbrow(out)
# Visualize word use across articles
plosword(list('regression', 'neural network'), vis = 'TRUE')

# plot through time
pl_t = plot_throughtime(terms = c('regression', 'neural network'), limit = 59)
pl_t = plot_throughtime(terms = dic, limit = 59)
```

Related paper

Topics over Time: A Non-Markov Continuous-Time Model of Topical Trends

Use graphical model and Gibbs sampling to find the topic trend.

Topic model

Download the Simply Statistics Github repo from here: https://github.com/simplystats/simplystats.github.io

Read in the text files from the _posts subdirectory of the resulting set of files. You will want to use the tm package for this. The functions you will need are "DirSource" and "Vcorpus".

Now look at the meta data for the 926th document using the "meta" command

Now use the "tidy" command to tidy up the documents and then unnest the tokens.

Remove the stopwords

Calculate the most frequent words using group_by, count, and arrange

Only keep words in this list of English words: https://github.com/dwyl/english-words/blob/master/words.txt.zip and remove the 20 most frequent words.

Cast the tidy object into a DocumentTermMatrix object.

Use the LDA command in the topicmodels package to fit a topic model using 3 and 10 topics.

Make a wordcloud of the top 20 words from each of these models. Can you "label" any of them.

```r
library(tm)
# file = DirSource(directory = "/Users/gege/Dropbox/graduate/DataScience/project/Data/
# # reut21578 <- DirSource(system.file("texts", "crude", package = "tm"))
#   VCorpus() %>%
#   DocumentTermMatrix(control = list(removePunctuation = TRUE,
#                                      removeNumbers = TRUE,
#                                      stopwords = FALSE))
#
# meta(file[[926]])
# inspect(file[[926]])
#
# library("broom")
# tidy(file[[4]])
# tidy(as.data.frame(as.matrix(file)))
```

```
dsource = DirSource(directory = "/Users/gege/Downloads/PLoS_Comput_Biol")
# dsource = DirSource(directory = "/Users/gege/Dropbox/graduate/DataScience/project/Da
file = VCorpus(dsource, readerControl = list(reader = readPlain))
# %>% DocumentTermMatrix(control = list(removePunctuation = TRUE, removeNumbers = TRUE

library(dplyr)
library(tidytext)
tidyfile = tidy(file)
# tidyfile = tidy(as.data.frame(as.matrix(file)))

dic = scan("/Users/gege/Dropbox/graduate/DataScience/project/Data/words.txt", character

file_words = tidyfile %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)

fileunnest = tidyfile %>% unnest_tokens(word, text)

file_count = as.vector(table(file_words[, colnames(file_words) %in% "word"]))
file_count2 = rownames(as.matrix(table(file_words[, colnames(file_words) %in% "word"]))
```

```r
word_freq = file_count2[order(file_count, decreasing = T)][1:20]

wholevol = unique(fileunnest$word)
file_word = file_words[file_words$word %in% dic, ]
antistopword = unique(file_word$word)

anostopword = union(setdiff(wholevol, antistopword), word_freq)
```

see how the appearance of a word changes over time:

```r
library("stringr")
library("tidyr")
file_word$Year = as.double(str_split(file_word$id, pattern = fixed("_"), simplify = T)[
file_freq = file_word[,colnames(file_word) %in% c("word", "Year")] %>%
  count(Year, word) %>%
  ungroup() %>%
  complete(Year, word, fill = list(n = 0)) %>%
  group_by(Year) %>%
  mutate(year_total = sum(n),
         percent = n / year_total) %>%
  ungroup()
```

For example, we can use the broom package to perform logistic regression on each word.

```r
f_freq = file_freq[file_freq$percent > 0, ]
models = f_freq %>%
  group_by(word) %>%
  filter(sum(n) > 50) %>%
  do(tidy(glm(cbind(n, year_total - n) ~ Year, .,
              family = "binomial"))) %>%
  ungroup() %>%
  filter(term == "Year")

library(ggplot2)
```

```r
models %>%
  mutate(adjusted.p.value = p.adjust(p.value)) %>%
  ggplot(aes(estimate, adjusted.p.value)) +
  geom_point() +
  scale_y_log10() +
  geom_text(aes(label = word), vjust = 1, hjust = 1,
            check_overlap = TRUE) +
  xlab("Estimated change over time") +
  ylab("Adjusted p-value")
```

We can also use the ggplot2 package to display the top 6 terms that have changed in frequency over time.

Report

Introduction:

PLoS (Public Library of Science) is a nonprofit Open Access publisher, innovator and advocacy organization with a mission to accelerate progress in science and medicine by leading a transformation in research communication. The data analysis project is to find out the most common statistical techniques in all published PLoS papers and the trends over the last 10-15 years.

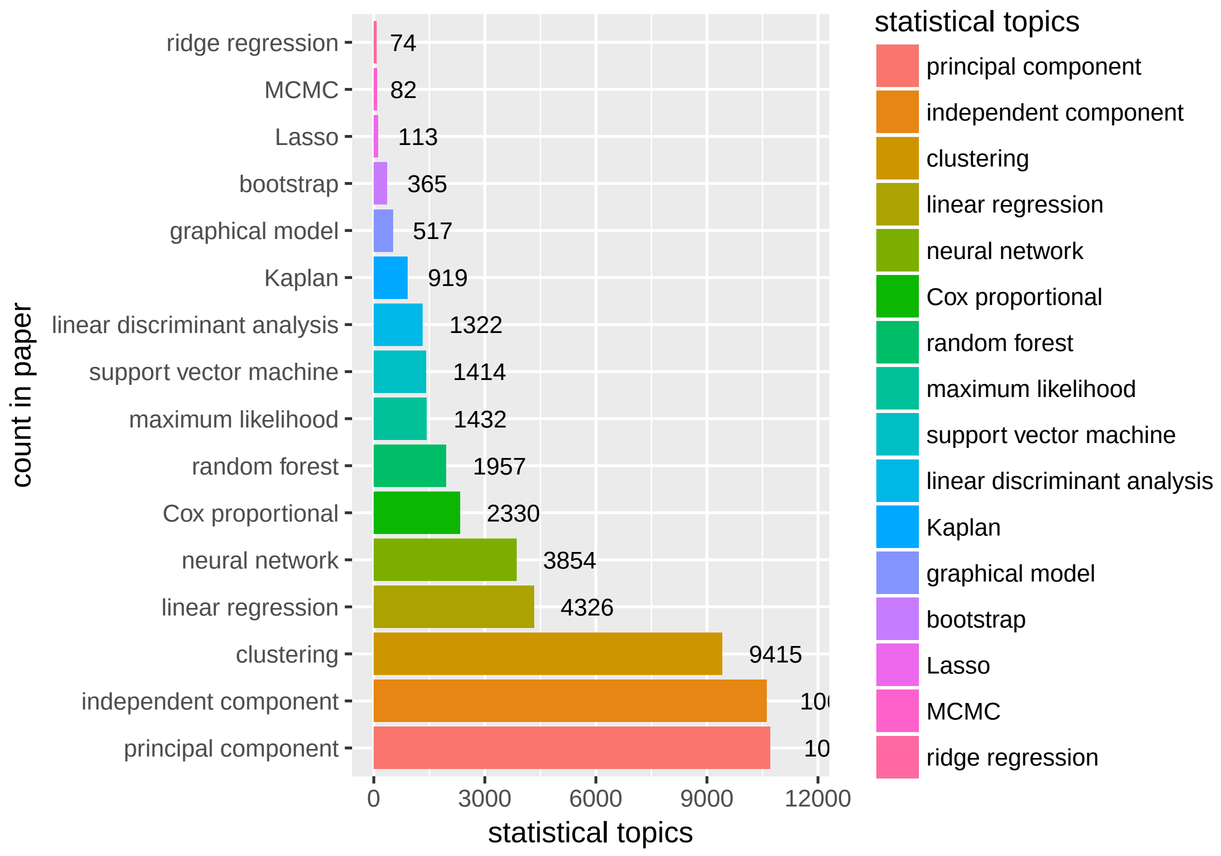
Data Preparation

The website has an API tool for finding related articles. ALM API can print out information about articles we are interesed in, such as author, publication date, abstract, etc. There is an R package “rplos”, which contains functions that can be used easily to look at article matrics.

Besides the “rplos” R package for summary statistics, we want to conduct text and data mining on our own by obtaining entire corpus. The raw data are from “Documentation: Text and Data Mining” page. As bulk downloading of article HTML/PDF/XML is discouraged, we use the “PMC OA Bulk Download FTP” to obtain research articles from multiple journals. The files are organized by topics alphabetically. We choose the folder named as “Biostatistics” as the one for model development. There are 52 documents in total. We use “LDA” (Latent Dirichlet Allocation) in R package “topicmodels” together with text mining commands in R package “tm”.

Method

We use “rplos” to do some exploritory data analysis. To use regular expression for further search, we create a dictionary for common statistical methods. We use “The Elements of Statistical Learning - Data Mining, Inference, and Prediction” second edition, by Trevor Hastie, Robert Tibshirani, Jerome Friedman, as a reference. The words from the dictionary are counted and the barplots below show the usage frequency of each word.



Latent Dirichlet Allocation (LDA) is the common algorithm used for distinguishing topics by text mining. R has several packages based on NLP to solve this kind of problem.

The articles from PLoS "Biostatistics" folder are text files. After reading the file using "DirSource" and "Vcorpus" from "tm" package, we tidy the data using "tidy" command from package "tidyr", remove the stopwords and punctuations to create a corpus. Only those words that are included in this list of English words: https://github.com/dwyl/english-words/blob/master/words.txt.zip, remain in the corpus. After calculating out the frequency for each word in every document, we implement LDA to find out possible topics. The initial value of topics are set to be 3 - 10. We need to find an appropriate method to determine how many topics should be left in the model and determine the label.

Another important part about this project is to find out the topics change over time. We first explore the word change over time, then use those words to see if there are links to models. An alternative way is to create a new variable: year, and group the word by year to see the trend. We can add "date" as a variable in LDA model. The next step would be to modify the Gibbs sampling procedure when considering time as an updated variable across sampling.