Tutorial (week 2) A: Interactive R Notebook

This is an *Interactive* R Markdown (http://rmarkdown.rstudio.com) Notebook. It generates an HTML notebook that would allow users to interactively explore your analysis results.

We will use the presidential inaugural speech word clouds as examples.

#Step 0 - Install and load libraries

```
packages.used=c("tm", "wordcloud", "RColorBrewer",
                "dplyr", "tydytext")

# check packages that need to be installed.
packages.needed=setdiff(packages.used,
                        intersect(installed.packages()[,1],
                                  packages.used))
# install additional packages
if(length(packages.needed)>0){
  install.packages(packages.needed, dependencies = TRUE,
                   repos='http://cran.us.r-project.org')
}

library(tm)
library(wordcloud)
library(RColorBrewer)
library(dplyr)
library(tidytext)
```

This notebook was prepared with the following environmental settings.

```
print(R.version)
```

```
##                _
## platform       x86_64-apple-darwin15.6.0
## arch           x86_64
## os             darwin15.6.0
## system         x86_64, darwin15.6.0
## status
## major          3
## minor          4.3
## year           2017
## month          11
## day            30
## svn rev        73796
## language       R
## version.string R version 3.4.3 (2017-11-30)
## nickname       Kite-Eating Tree
```

Step 1 - Read in the speeches

```
folder.path="/Users/fanerror/GitHub/Columbia/ADS_Teaching/Tutorials/wk2-TextMining/data/
fulltext/"
speeches=list.files(path = folder.path, pattern = "*.txt")
prex.out=substr(speeches, 6, nchar(speeches)-4)

ff.all<-Corpus(DirSource(folder.path))
```

#Step 2 - Text processing

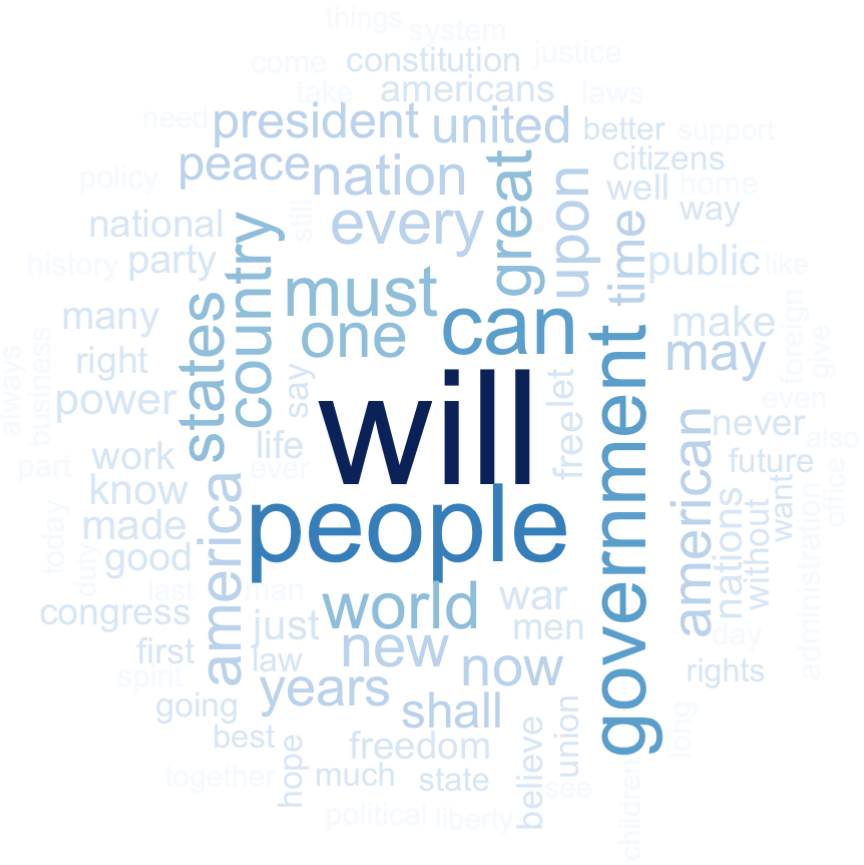
See Basic Text Mining in R (https://rstudio-pubs-static.s3.amazonaws.com/31867_8236987cf0a8444e962ccd2aec46d9c3.html) for a more comprehensive discussion.

For the speeches, we remove extra white space, convert all letters to the lower case, remove stop words (https://github.com/arc12/Text-Mining-Weak-Signals/wiki/Standard-set-of-english-stopwords), removed empty words due to formatting errors, and remove punctuation. Then we compute the Document-Term Matrix (DTM) (https://en.wikipedia.org/wiki/Document-term_matrix).

```
ff.all<-tm_map(ff.all, stripWhitespace)
ff.all<-tm_map(ff.all, content_transformer(tolower))
ff.all<-tm_map(ff.all, removeWords, stopwords("english"))
ff.all<-tm_map(ff.all, removeWords, character(0))
ff.all<-tm_map(ff.all, removePunctuation)

tdm.all<-TermDocumentMatrix(ff.all)

tdm.tidy=tidy(tdm.all)

tdm.overall=summarise(group_by(tdm.tidy, term), sum(count))
```

#Step 3 - Inspect an overall wordcloud

```
wordcloud(tdm.overall$term, tdm.overall$`sum(count)`,
          scale=c(5,0.5),
          max.words=100,
          min.freq=1,
          random.order=FALSE,
          rot.per=0.3,
          use.r.layout=T,
          random.color=FALSE,
          colors=brewer.pal(9,"Blues"))
```



#Step 4 - compute TF-IDF weighted document-term matrices for individual speeches. As we would like to identify interesting words for each inaugural speech, we use TF-IDF (https://en.wikipedia.org/wiki/Tf%E2%80%93idf) to weigh each term within each speech. It highlights terms that are more specific for a particular speech.

```
dtm <- DocumentTermMatrix(ff.all,
                          control = list(weighting = function(x)
                                             weightTfIdf(x,
                                                         normalize =FALSE),
                                         stopwords = TRUE))
ff.dtm=tidy(dtm)
```

#Step 5- Interactive visualize important words in individual speeches

```
library(shiny)

shinyApp(
    ui = fluidPage(
      fluidRow(style = "padding-bottom: 20px;",
        column(4, selectInput('speech1', 'Speech 1', speeches, selected=speeches[5])),
        column(4, selectInput('speech2', 'Speech 2', speeches, selected=speeches[9])),
        column(4, sliderInput('nwords', 'Number of words', 3, min = 20,
                              max = 200, value=100, step = 20))
      ),
      fluidRow(
        plotOutput('wordclouds', height = "400px")
      )
    ),

    server = function(input, output, session) {
      # Combine the selected variables into a new data frame
      selectedData <- reactive({
        list(dtm.term1=ff.dtm$term[ff.dtm$document==as.character(which(speeches == input
$speech1))],
             dtm.count1=ff.dtm$count[ff.dtm$document==as.character(which(speeches == inp
ut$speech1))],
             dtm.term2=ff.dtm$term[ff.dtm$document==as.character(which(speeches == input
$speech2))],
             dtm.count2=ff.dtm$count[ff.dtm$document==as.character(which(speeches == inp
ut$speech2))])
      })

      output$wordclouds <- renderPlot(height = 400, {
        par(mfrow=c(1,2), mar = c(0, 0, 3, 0))
        wordcloud(selectedData()$dtm.term1,
                  selectedData()$dtm.count1,
              scale=c(4,0.5),
              max.words=input$nwords,
              min.freq=1,
              random.order=FALSE,
              rot.per=0,
              use.r.layout=FALSE,
              random.color=FALSE,
              colors=brewer.pal(10,"Blues"),
            main=input$speech1)
        wordcloud(selectedData()$dtm.term2,
                  selectedData()$dtm.count2,
              scale=c(4,0.5),
              max.words=input$nwords,
              min.freq=1,
              random.order=FALSE,
              rot.per=0,
              use.r.layout=FALSE,
              random.color=FALSE,
              colors=brewer.pal(10,"Blues"),
            main=input$speech2)
      })
```

```
    },

    options = list(height = 600)
)
```

Speech 1

farewellGeorgeWashington-NA.txt

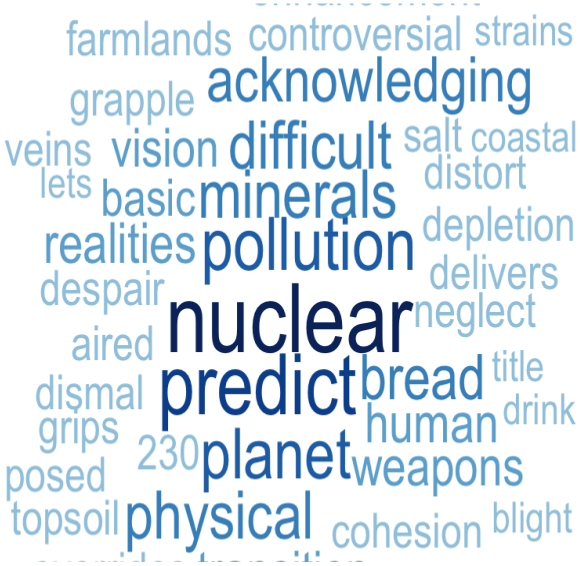
Speech 2

farewellJimmyCarter-NA.txt

Number of words

20 100 200

20 40 60 80 100 120 140 160 180 200



Further readings

- Text mining with `tidytext` (http://tidytextmining.com/).
- Basic Text Mining in R (https://rstudio-pubs-static.s3.amazonaws.com/31867_8236987cf0a8444e962ccd2aec46d9c3.html)