Course: VISUAL ANALYTICS FOR POLICY AND MANAGEMENT

Prof. José Manuel Magallanes, PhD

- Visiting Professor of Computational Policy at Evans School of Public Policy and Governance, and eScience Institute Senior Data Science Fellow, University of Washington.
- Professor of Government and Political Methodology, Pontificia Universidad Católica del Perú.

Text Data

Text can be used for plotting. The plots however require some treatment to the text because words can have several inflections. In this session, you will see the construction of word clouds. These plots represent a variation of bar plots for categories, but are attractive to the eye to show text.

1. Get the text:

Let me get a data frame with texts from some tweets:

```
link1="https://github.com/EvansDataScience/VAforPM_Text/"
link2="raw/main/trumps.csv"
trumpLink=paste0(link1,link2)
allTweets=read.csv(trumpLink ,stringsAsFactors = F)
```

2. Make some selection:

This data frame has some columns that allow subsetting. In this case, I will keeping tweets that are not retweets.

```
DTtweets=allTweets[allTweets$is_retweet==FALSE ,] #no
row.names(DTtweets)=NULL

#currently:
head(DTtweets)
```

```
##            created_at
## 1 2020-08-13 23:26:50
## 2 2020-08-13 23:23:26
## 3 2020-08-13 23:23:25
## 4 2020-08-13 18:59:29
## 5 2020-08-13 18:59:28
## 6 2020-08-13 18:59:26
##
## 1 .@DonYoungAK really produces for Alaska. He is an incredible Congressman who loves his State and w
## 2
## 3                                                                                .@CynthiaMLummis is a friend of m
## 4
## 5
## 6
##   is_retweet favorite_count retweet_count Hour Day       Date
## 1      FALSE          18714          5305   23   5 2020-08-13
## 2      FALSE          14946          3490   23   5 2020-08-13
## 3      FALSE          32586          8584   23   5 2020-08-13
```

```
## 4      FALSE          20131          5303   18   5 2020-08-13
## 5      FALSE          22491          5979   18   5 2020-08-13
## 6      FALSE          23984          6427   18   5 2020-08-13
```

3. Turn the text into words.

This process, also known as tokenization, will produce a simpler element from the input text, in this case words:

```
library(tidytext)
library(magrittr)
DTtweets_Words = DTtweets %>%
                 unnest_tokens(output=EachWord, # column created
                               input=text,# input column from DTtweets
                               token="words") # level of unnesting

head(DTtweets_Words,10) # notice 'EachWord'
```

```
##              created_at is_retweet favorite_count retweet_count Hour Day
## 1   2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.1 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.2 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.3 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.4 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.5 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.6 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.7 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.8 2020-08-13 23:26:50      FALSE          18714          5305   23   5
## 1.9 2020-08-13 23:26:50      FALSE          18714          5305   23   5
##           Date    EachWord
## 1   2020-08-13  donyoungak
## 1.1 2020-08-13      really
## 1.2 2020-08-13    produces
## 1.3 2020-08-13         for
## 1.4 2020-08-13      alaska
## 1.5 2020-08-13          he
## 1.6 2020-08-13          is
## 1.7 2020-08-13          an
## 1.8 2020-08-13  incredible
## 1.9 2020-08-13 congressman
```

You have these many 'words':

```
nrow(DTtweets_Words) # count of words
```

```
## [1] 3028
```

4. Getting rid of **common words**: These are know as the *STOP WORDS*:

```
# calling the file
data(stop_words)
# seeing some 'STOP WORDS'
head(stop_words)
```

```
## # A tibble: 6 x 2
##   word      lexicon
##   <chr>     <chr>
## 1 a         SMART
```

```
## 2 a's       SMART
## 3 able      SMART
## 4 about     SMART
## 5 above     SMART
## 6 according SMART
```

Then, we remove the stop words from the *EachWord* column:

```
library(dplyr)

 # The column 'word' from 'stop_words' will be compared # to the column 'EachWord' in 'DTtweets_Words'
DTtweets_Words = DTtweets_Words %>%anti_join(stop_words,
                                             by = c("EachWord" = "word"))

# You have these many rows now:

nrow(DTtweets_Words) # count of words
```

```
## [1] 1496
```

5. Compute **frequency** of each word:

Here, you are simply producing a frequency table. You could create a barplot with this.

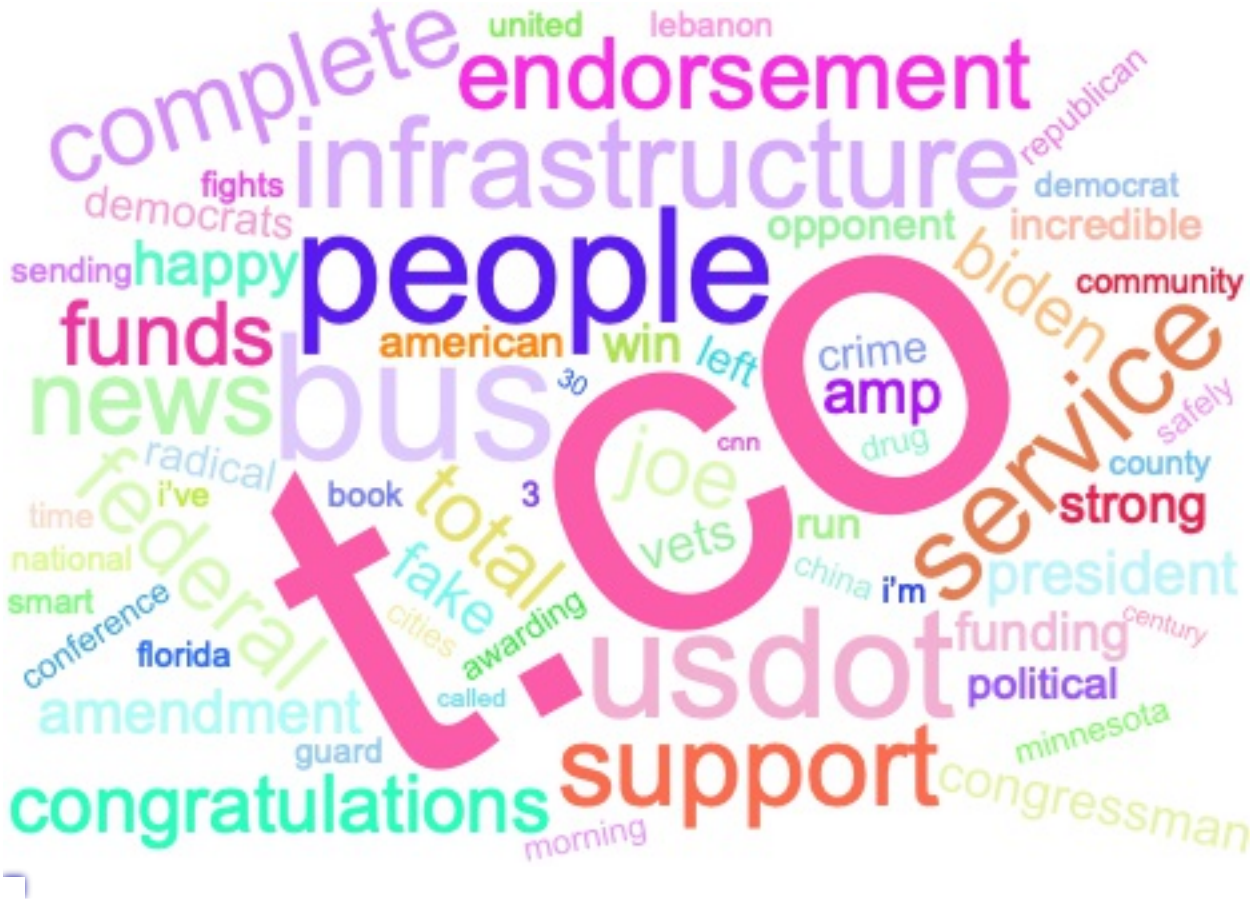
```
FTtrump = DTtweets_Words %>%dplyr::count(EachWord , sort = TRUE)
head(FTtrump)
```

```
##         EachWord  n
## 1          https 54
## 2           t.co 54
## 3            bus 20
## 4         people 18
## 5          usdot 17
## 6 infrastructure 14
```

6. Create a word cloud:

```
library(wordcloud2)

wc1=wordcloud2(data=FTtrump , size=1,minSize = 0,
               fontFamily = 'Arial', color='random-light', backgroundColor = "white",
               shape = 'circle') # option for shape are:
                                 # cardioid,diamond,triangle-forward,triangle,pentagon or star.
wc1
```



You can improve the cloud by getting rid of words that do not appear often and using your own color scale:

1. Getting rid of words with frequency below a threshold:

```
FTsub=FTtrump[FTtrump$n>4,]
```

2. Preparing a palette with the amount of colors desired:

```
library(RColorBrewer)
colorTones=max(FTsub$n)
paletteFun = colorRampPalette(c("orange","brown"))
palette=paletteFun(colorTones)[FTsub$n]
```

3. Recreating plot:

```
#This may not show in html.
#See: https://github.com/Lchiffon/wordcloud2/issues/65

wc2=wordcloud2(FTsub , color=palette)
wc2
```

Exercises:

1. Try this code instead of the previous one, and discuss the differences you see.

```
colorTones = length(unique(FTsub$n))
newColors = brewer.pal(9,"Reds")
paletteFun = colorRampPalette(newColors)
palette=paletteFun(colorTones)[FTsub$n]
```

2. Before step 3 above, do this cleaning processs and recreate the word cloud:

a. Getting rid of Emoticons. People use emoticons in their Tweets. Unless you translate their meaning, these are not to be analyzed. Use this code to get rid of them:

```
DTtweets$text=gsub("[^\x01-\x7F]", "", DTtweets$text)
```

b. Get rid of URLs. Generally, you do not need the URLs.

```
DTtweets$text=gsub("http\\S+\\s*","", DTtweets$text)
```

c. Special characters. Pay attention to symbols like &, >, or <, which may need to be replaced or eliminated like this:

```
DTtweets$text=gsub("&amp;", "and", DTtweets$text) #replaced
DTtweets$text=gsub("&lt;|&gt;", "", DTtweets$text) #eliminated
```

d. Get rid of users?. Evaluate if you should delete the mentions to other Twitter users, if you believe they

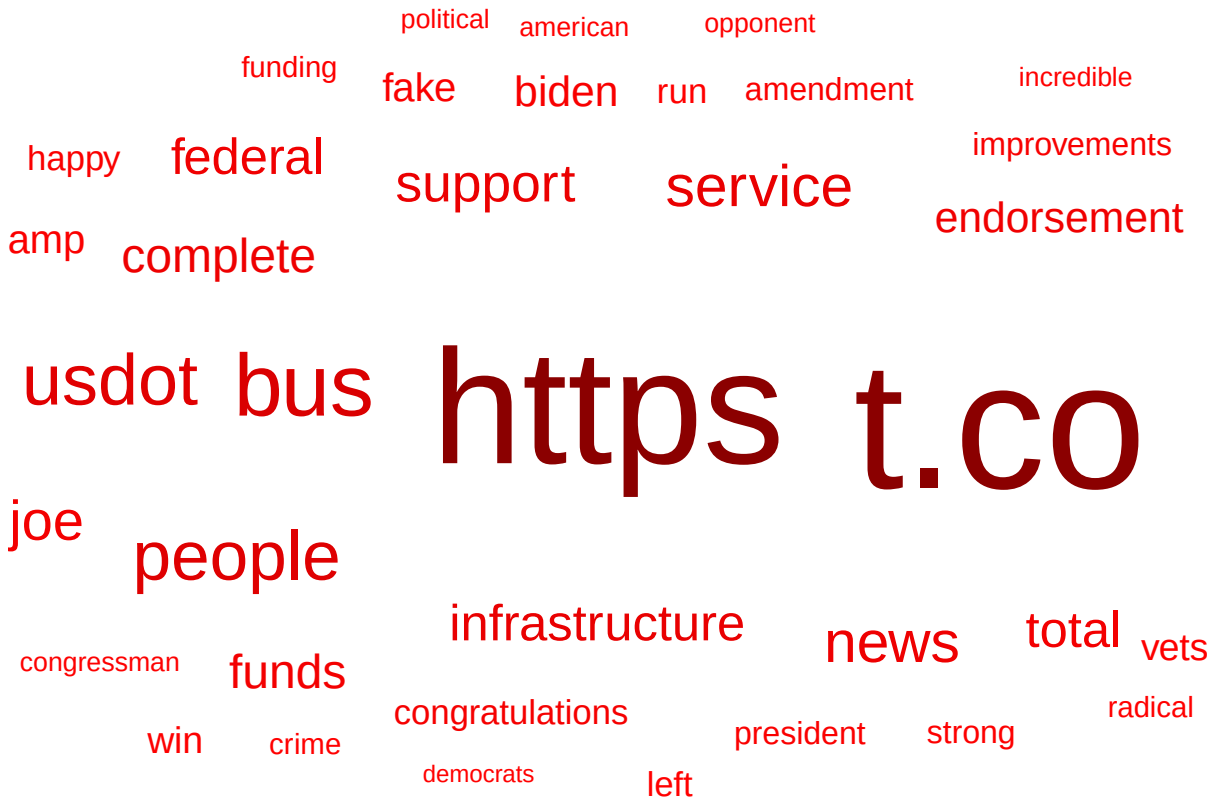
do not matter. Fir this exercise do NOT do this:

```
DTtweets$text=gsub("@\\w+", "", DTtweets$text)
```

e. Get rid of Hashtags? You can delete the hashtags, if you believe they do not matter:

```
DTtweets$text=gsub("#\\w+", "", DTtweets$text)
```

2. Try using ggwordcloud instead of *wordcloud2*. Write the code to produce the cloud below.

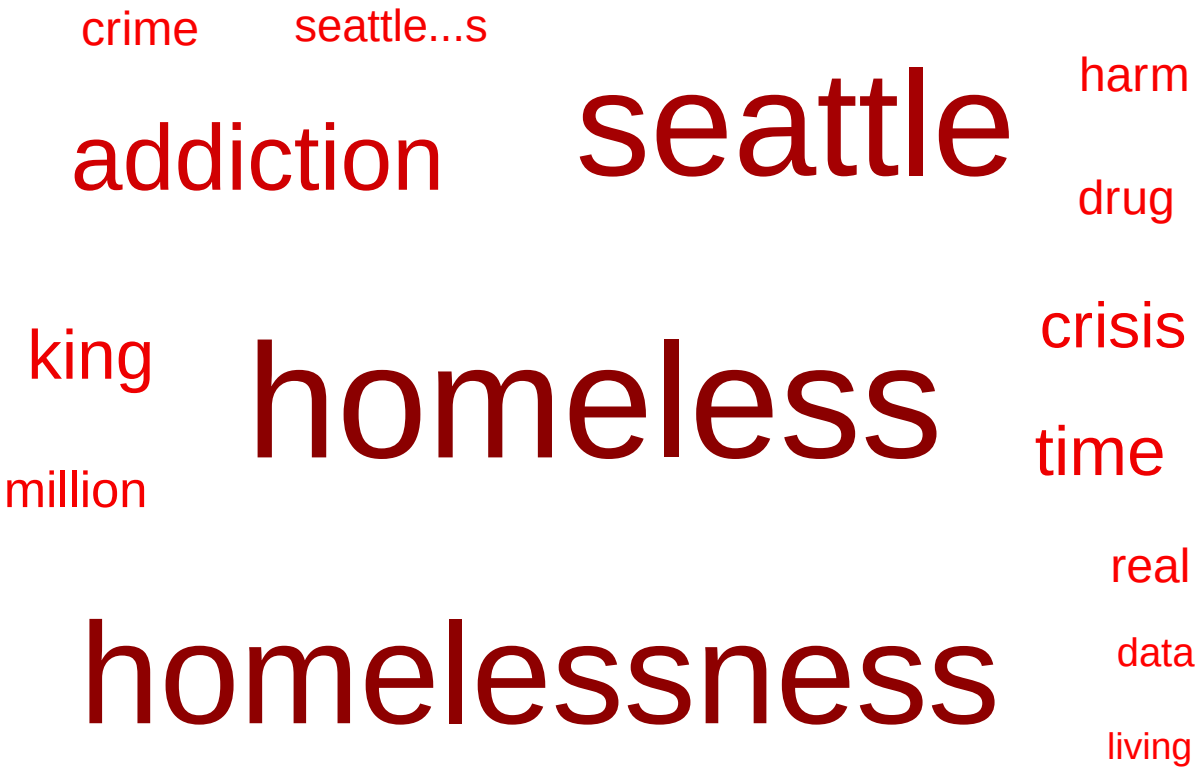


3. Create a new word cloud with the file 'sometext.txt':

```
otherText <- read.delim("sometext.txt",header = F)
head(otherText,2)
```

```
##
## 1 Seattle is under siege. Over the past five years, the Emerald City has seen an explosion of homeles
## 2                                                                                                     A
```

Write the code to produce the cloud below:


crime
seattle...s
addiction
seattle
harm
drug
king
homeless
crisis
million
time
real
homelessness
data
living