# *Working with Wordcloud*

Shraddha Dubey

November 2017

Lets start working with Wordcloud and we need some words to create a wordcloud. I picked up one of the novels of Charles Dickens - Tale of Two Cities. The book is available to download from gutenbergr package.

## 1 Libraries

Install and bring the following libraries to your workspace.

- **gutenbergr** : The gutenbergr package helps you download and process public domain works from the Project Gutenberg collection.

  Metadata for all Project Gutenberg works as R datasets, so that they can be searched and filtered.(**?**)

  - **gutenberg_download()** :downloads one or more works from Project Gutenberg by ID e.g., gutenberg_download(84) downloads the text of Frankenstein.
  - **gutenberg_metadata** : contains information about each work, pairing Gutenberg ID with title, author, language, etc
  - textbfgutenberg_authors : contains information about each author, such as aliases and birth/death year
  - **gutenberg_subjects**: contains pairings of works with Library of Congress subjects and topics

- **dplyr** :The dplyr package helps with data manipulation challenges.

  It provides simple "verbs", functions that correspond to the most common data manipulation tasks.It uses efficient backends, so we spend less time waiting for the computer.

  - *Dplyr* has following function for basic data manipulation:
  - *filter()* to select cases based on their values.
  - *arrange()* to reorder the cases.
  - *select()* and *rename()* to select variables based on their names.
  - *mutate()* and *transmute()* to add new variables that are functions of existing variables.
  - *summarise()* to condense multiple values to a single value.
  - *sample_n()* and *sample_frac()* to take random samples.

- *tidytext* : Its used for Text mining and sentiment analysis along with other tools like dplye and ggplot2

- *wordcloud* : A wordcloud is handy tool to highlight the most commonly cited words in a text using a quick visualization.

```
# download all the below mentioned packages first
#using the following command
#install.packages("package_name")
# install.packages("gutenbergr")
library(gutenbergr)
# install.packages("dplyr")
library(dplyr)
# install.packages("knitr")
library(knitr)
# install.packages("tidytext")
library(tidytext)
# install.packages("wordcloud")
library(wordcloud)
library(wordcloud2)
# install.packages("ggplot2")
library(ggplot2)
```

# 2 Download book

The first step towards making a wordcloud is getting text data. We are downloading 'A Tale of Two Cities' from gutenberg_works

```
gutenberg_works(title=='A Tale of Two Cities')
```

```
## # A tibble: 1 x 8
##   gutenberg_id                title           author gutenberg_author_id
##          <int>                <chr>            <chr>               <int>
## 1           98 A Tale of Two Cities Dickens, Charles                  37
## # ... with 4 more variables: language <chr>, gutenberg_bookshelf <chr>,
## #   rights <chr>, has_text <lgl>
```

```
# we can used the gutenberg_id to download
# the book into a dataframe

two_cities<-gutenberg_download(98)
two_cities[1:10,]
```

```
## # A tibble: 10 x 2
##    gutenberg_id                            text
##           <int>                           <chr>
##  1           98             A TALE OF TWO CITIES
##  2           98
##  3           98 A STORY OF THE FRENCH REVOLUTION
##  4           98
##  5           98               By Charles Dickens
##  6           98
##  7           98
##  8           98                        CONTENTS
##  9           98
## 10           98
```

# 3 Unnest the words

As you can notice in the above section, the text column holds each sentence of the novel.
Lets break it down into words using dplyr[1] function *unnest_tokens()*.
unnest_tokens() takes two parameters, first is the name of the resultant column and second is the name of source column which we want to unnest.

```
# break the lines into words and store into dataframe as two_cities
two_cities<-two_cities%>%
  unnest_tokens(word,text)

two_cities[1:20,]

## # A tibble: 20 x 2
##    gutenberg_id       word
##           <int>      <chr>
##  1           98          a
##  2           98       tale
##  3           98         of
##  4           98        two
##  5           98     cities
##  6           98          a
##  7           98      story
##  8           98         of
##  9           98        the
## 10           98     french
## 11           98 revolution
## 12           98         by
## 13           98    charles
## 14           98    dickens
## 15           98   contents
## 16           98       book
## 17           98        the
## 18           98      first
## 19           98   recalled
## 20           98         to
```

[1]ref at https://cran.r-project.org/web/packages/dplyr/vignettes/dplyr.html

# 4 Sentiment Analysis

Now that we have all the words from the novel, lets evaluate the sentiments of those words.

For that we can import a sentiment lexicon 'nrc' from tidytext package [2]

```
two_cities$gutenberg_id<-NULL

# store all the sentiments into sent dataframe
sent<-get_sentiments('nrc')

# inner join of sent dataframe with two_cities
# will give us the respective sentiments for each word

two_cities<-inner_join(two_cities,sent)

## Joining, by = "word"

two_cities[1:10,]

## # A tibble: 10 x 2
##           word    sentiment
##          <chr>        <chr>
##  1        tale     positive
##  2  revolution        anger
##  3  revolution anticipation
##  4  revolution         fear
##  5  revolution     negative
##  6  revolution     positive
##  7  revolution      sadness
##  8  revolution     surprise
##  9        mail anticipation
## 10 preparation anticipation
```

[2] more details at https://cran.r-project.org/web/packages/tidytext/vignettes/tidytext.html

# 5 Plotting

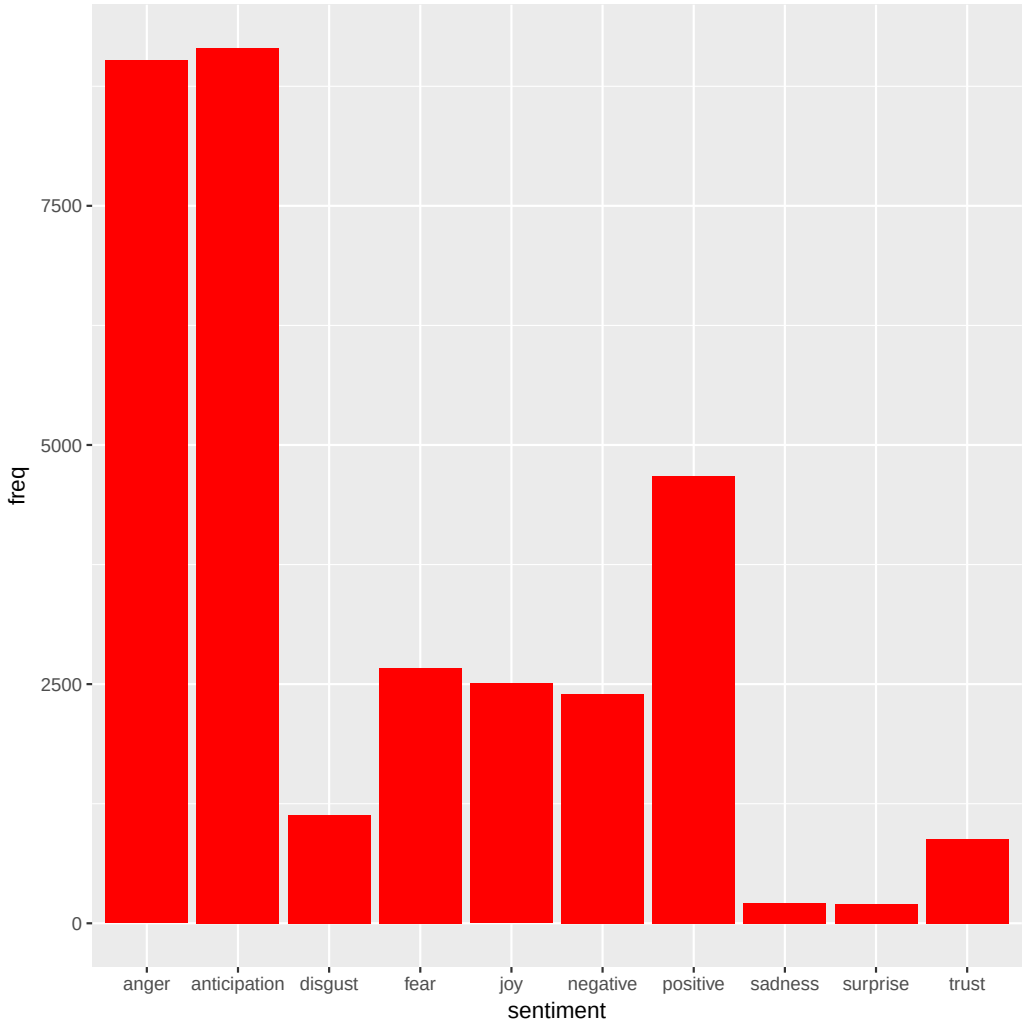
We have all the sentiments and words so lets plot the graph for the sentiments used in the novel using *ggplot2*.

```
# group the words by the sentiments

two_cities<-two_cities%>%
  group_by(word)%>%
  summarize(freq=n(),sentiment=first(sentiment))
two_cities[1:10,]

## # A tibble: 10 x 3
##           word  freq sentiment
##          <chr> <int>     <chr>
##  1     abandon     3      fear
##  2   abandoned    40     anger
##  3 abandonment     5     anger
##  4     ability     2  positive
##  5      abject     2   disgust
##  6   abolition     1  negative
##  7  abominable     6   disgust
##  8      abrupt     3  surprise
##  9     absence    24      fear
## 10      absent    10  negative

# plotting graph
ggplot()+
  geom_bar(data=two_cities,aes(x=sentiment,y=freq)
           ,stat = 'identity',fill='red')
```




# 6 wordcloud

```
wordcloud(two_cities$word,two_cities$freq,min.freq =50,
          colors = brewer.pal(6,'Dark2'),vfont=c("script","bold"))
```


good
doctor
tribunal
true
imprisonment
surprise
creature
hope
fancy
sweet
general
dark
mother
pretty
case
miserable
money
fellow
bear
usual
long
respect
lost
armed
wild
shoulder
attorney
remarkable
fortune
revolution
darkness
rising
gentleman
thought
delay
force
die
liberty
question
guard
punch
coach
top
young
journey
present
danger
fellow
church
pleasant
death
words
court
youth
horror
tree
leave
sin
dear
child
boy
feeling
black
public
terrible
safe
dead
advance
honest
dreadful
doubt
pray
ill
difficulty
smile
weight
stone
happy
misery
save
grave
fire
love
bank
broken
saint
father
time
confidence
promise
daughter
sun
kind
coming
cry
cap
hanging
worse
quiet
white
brother
finally
fear
precious
engaged
immediately
influence
lying
friend
bad
found
lord
god
citizen
pay
word
prisoner
guillotine
vengeance
grim
devil
prison
cruel