The Strange Case of Dr. Jekyll & Mr. Hyde in Wordclouds

Elizabeth Bross

November 6, 2017

Abstract

In this article we will construct two wordclouds, using the tidytext R package, for Robert Louis Stevenson's novel *The Strange Case of Dr. Jekyll & Mr. Hyde*. For those unfamiliar, *The Strange Case of Dr. Jekyll & Mr. Hyde* is a story set in Victorian era London about a series of odd coincidents bewtween two opposite characters: the kind Dr. Henry Jekyll and the evil Mr. Edward Hyde. As the story progresses, the reader becomes aware that Jekyll and Hyde are indeed the same person. The other characters are tasked with connecting occurances and eventually realize Dr. Jekyll transforms into Hyde by drinking a serum he invented. To capture the 'good versus evil' theme of the book, we will create two wordclouds using opposing sentiment categories.

1 The gutenbergr Package

The package for R, gutenbergr, that gives one electronic access to over 50,000 free books. The collection is made up of the world's great titles from throughout history and includes older works with expired copyrights[1]. To begin this project, install the required packages and download the text. The gutenberg_id number for *The Strange Case of Dr. Jekyll & Mr. Hyde* is 42.

```
library(tm)
library(tidytext)
library(dplyr)
library(knitr)
library(gutenbergr)
library(stringr)
library(wordcloud)
JandH<-gutenberg_download(42)
```

[1] you can find more information on Project Gutenberg here: https://www.gutenberg.org/

2 Data Preparation

We must first create a dataframe of all the words in the novel. This is easy to do with tidytext package. Use the unnest_tokens function to break down the words into a dataframe.

```
JandH_words<-JandH%>%
  unnest_tokens(word,text)
```

To determine which words we will look at, we will use those listed in the sentiment lexicon 'nrc'. The tidytext package provides four sentiment lexicons. The 'nrc' lexicon assign one of ten descriptions to important words in the English language. This technique eliminates the need to remove 'stopwords'[2] which are common, irrelevent words.

```
nrc<-get_sentiments('nrc')
unique(nrc$sentiment)

##  [1] "trust"        "fear"         "negative"     "sadness"
##  [5] "anger"        "surprise"     "positive"     "disgust"
##  [9] "joy"          "anticipation"

nrc_fear<-nrc%>%
  filter(sentiment %in% c('fear','negative',
                          'sadness','anger','disgust'))

nrc_happy<-nrc%>%
  filter(sentiment %in% c('trust','surprise',
                          'positive','joy','anticipation'))
```

Then, use the inner_join function to match the words in our newly created dataframe JandH_words with words provided in the nrc lexicon.

```
JandH_fear_words<-inner_join(nrc_fear,JandH_words)
JandH_happy_words<-inner_join(nrc_happy,JandH_words)
```

3 The Wordclouds

To begin creating the word coulds, use dplyr to calculate the frequency of words included in the JandH_fear_words dataframe. Be sure to create a new dataframe to make the creation of the final wordcloud easier.

[2] a list of stopwords is available in the R package tm.

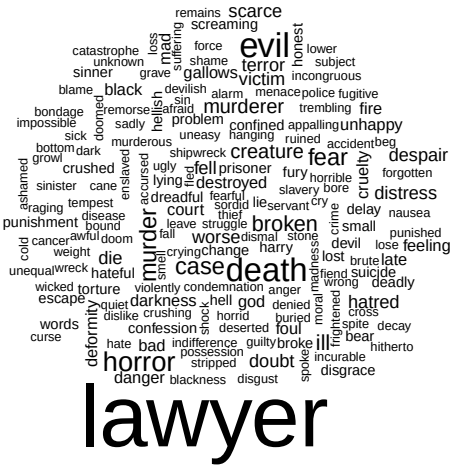
```
  JandH_fear_freq<-JandH_fear_words%>%
  group_by(word)%>%
  summarize(count=n())
```

Then repeat this step on the JandH_happy_words dataframe.

```
JandH_happy_freq<-JandH_happy_words%>%
  group_by(word)%>%
  summarize(count=n())
```

Finally, use the wordcloud() function to create the fear themed wordcloud. In context of *The Strange Case of Dr. Jekyll & Mr. Hyde*, you could call this the Hyde wordcloud. Set the min.freq to 5 to avoid overloading your wordcloud.

```
wordcloud(JandH_fear_freq$word,JandH_fear_freq$count,min.freq = 5)
```



Repeat with JandH_happy_freq to create your Jekyll wordcloud.

```
wordcloud(JandH_happy_freq$word,JandH_happy_freq$count,min.freq = 5)
```