CWRU DSCI351-451: Homework 4

Prof.:Roger French, TA: JiQi Liu

October 11, 2018

Contents


```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- tidyverse
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------------------------------------- tidyverse_confli
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tidytext)
library(dplyr)
library(ggplot2)
```

1.2 Text Mining of Song Lyrics

- Complete the given problems
- dplyr, ggplot, and pipelines are highly recommended
- We will be using the Tidytext package to aid with our text mining
- The dataset for this assignment is a collection of the information and lyrics from every top 100 billboard song since 1965

1.2.1 Problem 1

- Load in the data set
- Print the lyrics of the #4 song from 1988
- Show the top 20 artists with the most hits, which artist has the most total top 100 hits?

1.2.2 Problem 2

- Build a histogram of the amount of times artists appear on the top hits billboard, what does the trend look like, what does it suggest about 1 hit wonders?

1.2.3 Problem 3

- Lets do a lyrical comparison between 2 top artists, Elton John and Eminem
- First, filter the main data set to only the 2 given artists (you can make them seperate data frames or keep them together, your call)
- Use the unnest_tokens() function from tidytext to split the lyrics up so that each word has its own row
- Show the top 10 most commonly used words for each artist
- It should be clear that most of these are not significantly meaningful words and should be removed, we can remove them using the stop_words dataframe provided with the tidytext package (hint: look at dplyr anti_join())
- Remove them and show the top ten remaining words for each artist

```
data("stop_words")
```

1.2.4 Problem 4

- Build a word cloud of the lyrics for each atrist, find an R package that will help you with this
- Compare and contrast the word clouds
- Did the stop_words dataframe work well to remove non-meaningful words?

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

1.2.5 Problem 5

- The tidytext package gives us the ability to run sentiment analysis as well
- There are multiple sentiment methods available but let's just use positive and negative sentiment

```
bing <- get_sentiments("bing")
```

- What is the bing dataframe?
- Pull out the "positive" and "negative" words for each artist (hint: dplyr inner_join())
- For each song per artist, determine the next positive or negative sentiment, there are several ways to do this - the most straightforward being assign a 1 to a positive lyric and -1 to a negtive lyric then sum them together for each song
- Make a bar plot of the net sentiment of each song for each artist, make these plots high quality as well
  - Properly name the axes and title each plot
  - Color by whether the song is overall positive or negative
  - rotate the x-axis names so long song titles are legible
  - Arrange the bars in accending or descending order
- What differences do you notice between the 2?
- Based on what you might know about some of these songs, do you think the sentiment analysis gives a good indication of the positive of negative tone for each song or not?
- If you're interested, play around with this data set, see if your favorite artist is in here and see if you can find anything else