# nomin-amar-post1

## Introduction/Motivation:

Words mean something?

Analysts often learn to deal with tabular numeric data, but with the unimaginably fast rates of generating and collecting data nowadays, much of the data is non-tabular, text heavy, and unstructured. To work with this kind of data, text mining and language processing began to boom. Text mining involves using methods in information retrieval, statistics, and machine learning fields. As we know, one of the advantages of using R in data analysis is the fact that R is extensible with hundreds of packages that can be easily obtained to make your analysis easier. Using these packages as resources, the number of inferences we can make from a text data is vast. In practice, text mining is mainly used for :

- analyzing open-ended survey responses: especially in survey research such as in marketing, it is common to ask open-ended questions to explore a topic. With the answers provided, text mining can show trends in certain words pertaining to a certain product's pros and cons.
- automatic processing of messages, emails: it is possible to filter through emails to remove junk emails using text mining.
- analyzing warranty or insurance claims, diagnostic interviews : text mining allows filtering through claims to find the most common complaints and problems.
- investigating competitors by mining through their websites: you are able to quickly determine the most important terms and features that are described.
- sentiment analysis: analyzing movie reviews for approximating how favorable the review is for a movie.
- national security and intelligence: many software packages are made to monitor plain and text sources such as internet, blogs, and etc for national security purposes.
- social media mining: filtering through twitter, facebook, and etc to find correlation and/or draw inferences. All of these applications have a common purpose of "numerizing" text data to draw reasonable inferences from a given question/problem.

## Background:

The dilemma of exploiting and analyzing large unstructured and text heavy data has been around for decades. This dilemma and the potential of text mining have been recognized as early as 1958. Furthermore, it is described in the October 1958 issue of IBM Journal as, "…utilize data-processing machines for auto-abstracting and auto-encoding of documents and for creating interest profiles for each of the 'action points' in an organization. Both incoming and internally generated documents are automatically abstracted, characterized by a word pattern, and sent automatically to appropriate action points."

## Demonstration:

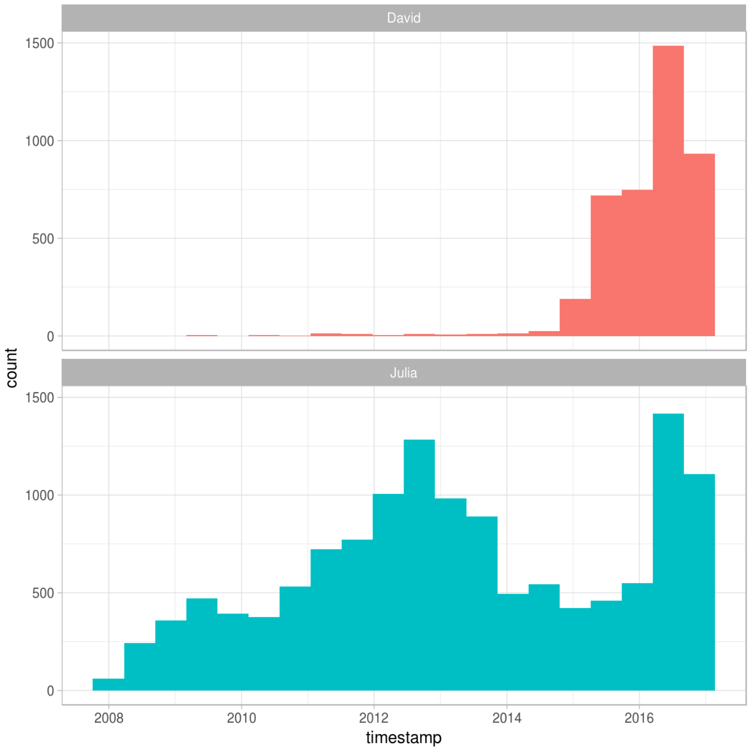
Among many useful packages in R that can be utilized in text mining. One useful package developed in 2016 by Silge and Robinson is called "tidytext." This package allows easier and more effective way to text mine. Additionally and importantly, it works well with the already present and available tools in use. The objective of this package is to turn individual words into data frames; hence, allowing easier manipulation.

Many people and industry are interested in words exchanged via social media. This demonstration will include a demonstration of text mining and analysis done on the two author's (Silge and Robinson) twitter archive.

## Getting the data and disribution of tweets

This step is always a necessity and the first step in any analysis. An individual is able to download their own Twitter archive following the directions on how. This step utilizes other packages, lubridate, ggplot2,dply,and readr.

```
library(lubridate)
library(ggplot2)
library(dplyr)
library(readr)

tweets_julia <- read_csv("data/tweets_julia.csv")
tweets_dave <- read_csv("data/tweets_dave.csv")
tweets <- bind_rows(tweets_julia %>%
                      mutate(person = "Julia"),
                    tweets_dave %>%
                      mutate(person = "David")) %>%
  mutate(timestamp = ymd_hms(timestamp))

ggplot(tweets, aes(x = timestamp, fill = person)) +
  geom_histogram(position = "identity", bins = 20, show.legend = FALSE) +
  facet_wrap(~person, ncol = 1)
```



As you can see from the histograms, David and Julia joined twitter around the same time. However, David was not active on twitter for about 5 five years. Overall, Julia tweeted approximately 4 times as many as David.

## Word Frequencies

Now that we have an overview of what the distribution looks like, a basic manipulation we can do is figure out the frequencies for each word Julia and Davis use. Intuitively speaking, in text analysis, words such as "which, this, that, and etc" would not add much to the inference we are trying to draw. Hence, these "stop words" are eliminated, and unnest_tokens() is used to make a tidy data frame of all the words in the tweets. To further clean our data, only the tweets are extracted, and the retweets are eliminated. Additionally, remove links and clean out some words and characters are intuitively not helpful in analysis.

```
library(tidytext)
library(stringr)

replace_reg <- "https://t.co/[A-Za-z\\d]+|http://[A-Za-z\\d]+|&amp;|&lt;|&gt;|RT|https"
unnest_reg <- "([^A-Za-z_\\d#@']|'(?![A-Za-z_\\d#@]))"
tidy_tweets <- tweets %>%
  filter(!str_detect(text, "^RT")) %>%
  mutate(text = str_replace_all(text, replace_reg, "")) %>%
  unnest_tokens(word, text, token = "regex", pattern = unnest_reg) %>%
  filter(!word %in% stop_words$word,
         str_detect(word, "[a-z]"))
```

After all the tidying is done, we can figure out the word frequencies for each person:

```
frequency <- tidy_tweets %>%
  group_by(person) %>%
  count(word, sort = TRUE) %>%
  left_join(tidy_tweets %>%
              group_by(person) %>%
              summarise(total = n())) %>%
  mutate(freq = n/total)

frequency
```

```
## Source: local data frame [20,736 x 5]
## Groups: person [2]
##
##    person           word     n total        freq
##     <chr>          <chr> <int> <int>       <dbl>
## 1   Julia           time   584 74572 0.007831358
## 2   Julia    @selkie1970   570 74572 0.007643620
## 3   Julia       @skedman   531 74572 0.007120635
## 4   Julia            day   467 74572 0.006262404
## 5   Julia           baby   408 74572 0.005471222
## 6   David @hadleywickham   315 20161 0.015624225
## 7   Julia           love   304 74572 0.004076597
## 8   Julia   @haleynburke   299 74572 0.004009548
## 9   Julia          house   289 74572 0.003875449
## 10  Julia        morning   278 74572 0.003727941
## # ... with 20,726 more rows
```

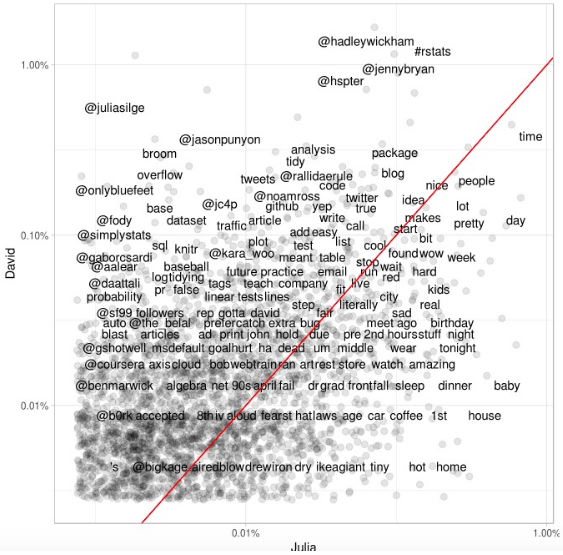
We can take it further by plotting these words and the frequencies for each person:

```
library(tidyr)

frequency <- frequency %>%
  select(person, word, freq) %>%
  spread(person, freq) %>%
  arrange(Julia, David)

frequency
```

```
## # A tibble: 17,640 × 3
##                word        David        Julia
##               <chr>        <dbl>        <dbl>
## 1                's 4.960071e-05 1.340986e-05
## 2  @accidental__art 4.960071e-05 1.340986e-05
## 3       @alice_data 4.960071e-05 1.340986e-05
## 4        @alistaire 4.960071e-05 1.340986e-05
## 5       @corynissen 4.960071e-05 1.340986e-05
## 6     @jennybryan's 4.960071e-05 1.340986e-05
## 7           @jsvine 4.960071e-05 1.340986e-05
## 8     @lizasperling 4.960071e-05 1.340986e-05
## 9        @ognyanova 4.960071e-05 1.340986e-05
## 10       @rbloggers 4.960071e-05 1.340986e-05
## # ... with 17,630 more rows
```

Use geom_jitter() to not put big emphasis on the discreteness of the low frequency words and use check_overlap=TRUE so that only some will print to avoid overcrowdness.

```
library(scales)

ggplot(frequency, aes(Julia, David)) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.25, height = 0.25) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  geom_abline(color = "red")
```



Words that are close to the red line are used by both Julia and David in equal frequency, while the words that are located further from line correlate to words that are not used by both Julia and David.

Even though this manipulation was simple, we can already draw some inferences. You can tell from this analysis so far that Davis has exclusively used Twitter for professional purposes, while Julia used it more personally in the beginning until later in 2015 when she started using it professionally.

## Comparing word usage

Now that we have calculated word frequencies, we can see which words are more likely or less likely to appear on Julia and David's twitter. In order to do this, we can utilize the log odds ratio. But first, we can restrict our data set so it is easier to work with, and we can make inferences aobout data science since that is when Julia started her data science career. To do this: Use log odds ratio:

$$\text{log odds ratio} = \ln\left(\frac{\left[\frac{n+1}{\text{total}+1}\right]_{\text{David}}}{\left[\frac{n+1}{\text{total}+1}\right]_{\text{Julia}}}\right)$$

```
tidy_tweets <- tidy_tweets %>%
  filter(timestamp >= as.Date("2016-01-01"),
         timestamp < as.Date("2017-01-01"))
```

Use str_detect() to remove the twitter usernames.After this, we can only count the words that appear more than 10 times.

```
word_ratios <- tidy_tweets %>%
  filter(!str_detect(word, "^@")) %>%
  count(word, person) %>%
  filter(sum(n) >= 10) %>%
  ungroup() %>%
  spread(person, n, fill = 0) %>%
  mutate_if(is.numeric, funs((. + 1) / sum(. + 1))) %>%
  mutate(logratio = log(David / Julia)) %>%
  arrange(desc(logratio))
```

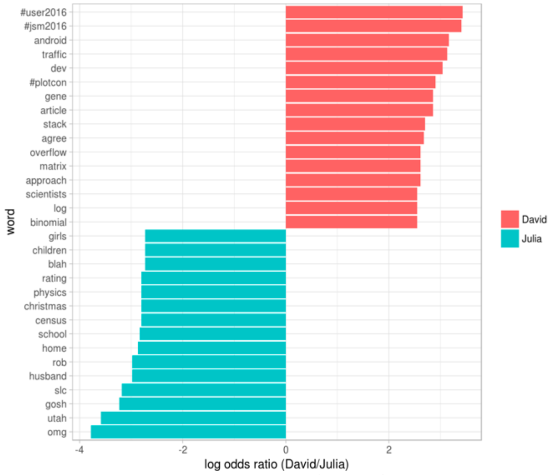
What are some words that appeared in equal frequency for both Julia and David?

```
word_ratios %>%
  arrange(abs(logratio))
```

```
## # A tibble: 377 × 4
##          word       David       Julia    logratio
##         <chr>       <dbl>       <dbl>       <dbl>
## 1         map 0.002321655 0.002314815 0.002950476
## 2       email 0.002110595 0.002083333 0.013000812
## 3        file 0.002110595 0.002083333 0.013000812
## 4       names 0.003799071 0.003703704 0.025423332
## 5     account 0.001688476 0.001620370 0.041171689
## 6         api 0.001688476 0.001620370 0.041171689
## 7    function 0.003376952 0.003240741 0.041171689
## 8  population 0.001688476 0.001620370 0.041171689
## 9         sad 0.001688476 0.001620370 0.041171689
## 10      words 0.003376952 0.003240741 0.041171689
## # ... with 367 more rows
```

Now which words mostly appear from Julia and which ones mostly appear from David?

```
word_ratios %>%
  group_by(logratio < 0) %>%
  top_n(15, abs(logratio)) %>%
  ungroup() %>%
  mutate(word = reorder(word, logratio)) %>%
  ggplot(aes(word, logratio, fill = logratio < 0)) +
  geom_col() +
  coord_flip() +
  ylab("log odds ratio (David/Julia)") +
  scale_fill_discrete(name = "", labels = c("David", "Julia"))
```



From here we can draw inferences about what each person likes and tweets the most about. ##Conclusions/Future of Text Mining: The applications of using computer science and statistics are limitless and powerful. An application that has been around since the 1950's and have became prominent in data analysis is text mining. This encompasses many industry examples such as automatic analysis of e-mails, texts, and insurance warranties. A commonly used powerful tool that analysts use for text mining is R. Furthermore, R has numerous current and new coming packages that make R more versatile and robust. One package that was specifically made for text mining by Julia Silge and David Robinson is 'tidytext.' This package is not only easy to use but it also works well with already existing functions on R. A simple example of text mining was explored here on the two authors' twitter archives. With just a common example of text mining, many inferences were drawn from the analysis. What is the future of text mining? what other industries and problems can we address with the given potentials of text mining?

## References:

http://tidytextmining.com/twitter.html

https://bitesizebio.com/23003/my-10-favorite-r-packages-and-the-cool-things-you-can-do-with-them/

https://www.predictiveanalyticstoday.com/text-analytics/

http://delivery.acm.org/10.1145/1090000/1089816/p1-kao.pdf?ip=136.152.142.142&id=1089816&acc=ACTIVE%20SERVICE&key=CA367851C7E3CE77%2E3158474DDFAA3F10%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=823592747&CFTOKEN=539776078

http://www.expertsystem.com/10-text-mining-examples/

http://www.statsoft.com/Textbook/Text-Mining#overview

https://eight2late.wordpress.com/2015/05/27/a-gentle-introduction-to-text-mining-using-r/