# How Did Donald Trump Win the Presidency?

*Fan Gong*

*2017/9/16*

## Overview

---

We all know that **Donald Trump** is the current president of the United States, in office since January 20, 2017. We also know that he has gone through a very tough campaign against with Hillary Clinton.

So, one question keep coming to my mind: How did Donald Trump win this presidency?

I was lucky enough to see the whole campaign between Donald Trump and Hillary Clinton. And here is the picture I took from the election night at Javits Convention Center.

Not to talk about the specific political dissidence between Trump and Hillary, let me start to find the answers through analyzing their speeches.

---

## Loading Packages

```
library(tm)
library(dplyr)
library(RColorBrewer)
library(wordcloud)
library(ggplot2)
library(qdap)
library(syuzhet)
library(gplots)
library(reshape2)
library(factoextra)
library(koRpus)
library(gridExtra)
library(ggthemes)
```

---

## Part One: Word Analysis

In this project, I use twenty speeches' transcripts as my datasets, ten from Donald Trump and ten from Hillary Clinton. I will first focus on identifying the word usage differences between them. Basically, there are three sub-analysis in this part:

- Frequency Analysis
- Keyness Analysis
- Readability Analysis

### Frequency Analysis

Word Frequency Analysis is always an important part in text mining. Since it gives us an vary general pattern about what kind of words this person are very likely to use.

### Data Pre-processing

- Load the Corpus
- Utilize `tm` package to clean the data
- Create a Document Term Matrix

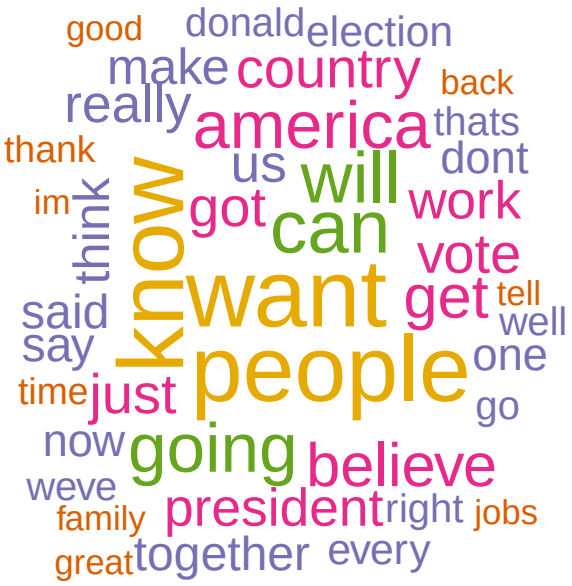
```
#specify the relevant path
docs.a = Corpus(DirSource("../data/Clinton-Trump Corpus/Clinton"))
docs.b = Corpus(DirSource("../data/Clinton-Trump Corpus/Trump"))


#process the files in the first corpus
docs.a = docs.a %>%
  tm_map(removeNumbers) %>%
  tm_map(tolower) %>%
  tm_map(PlainTextDocument) %>%
  tm_map(removePunctuation) %>%
  tm_map(tolower) %>%
  tm_map(removeNumbers) %>%
  tm_map(stripWhitespace) %>%
```

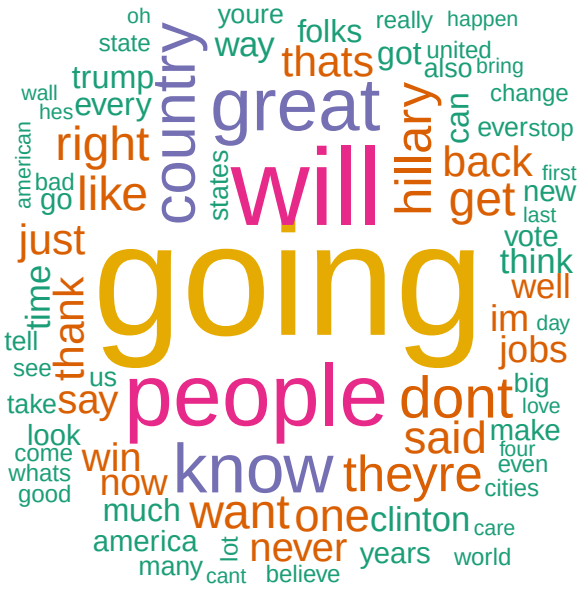
```
  tm_map(removeWords, c('s', 't', 're', 've', 'm')) %>% #Remove some unreasonable words
  tm_map(removeWords, stopwords("english"))


#process the files in the second corpus
docs.b = docs.b %>%
  tm_map(removeNumbers) %>%
  tm_map(tolower) %>%
  tm_map(PlainTextDocument) %>%
  tm_map(removePunctuation) %>%
  tm_map(tolower) %>%
  tm_map(removeNumbers) %>%
  tm_map(stripWhitespace) %>%
  tm_map(removeWords, c('s', 't', 're', 've', 'm')) %>%
  tm_map(removeWords, stopwords("english"))

#create a term matrix
dtm.a = DocumentTermMatrix(docs.a, control=list(wordLengths=c(1,1000)))
dtm.a = removeSparseTerms(dtm.a, 0.2)

wf.a = data.frame(word = colnames(dtm.a), freq=colSums(as.matrix(dtm.a))) %>%
  arrange(desc(freq))

dtm.b = DocumentTermMatrix(docs.b, control=list(wordLengths=c(1,1000)))
dtm.b = removeSparseTerms(dtm.b, 0.2)

wf.b = data.frame(word = colnames(dtm.b), freq=colSums(as.matrix(dtm.b))) %>%
  arrange(desc(freq))
```

### Make a Wordcloud

Based on the document-term matrix I have just created, we could highlight the most frequently used keywords in a texts by making a wordcloud.

```
#WordCloud for Clinton's Speech
par(mfrow = c(1,2))
set.seed(1)
colors = brewer.pal(6, "Dark2")
wordcloud(wf.a$word, wf.a$freq, min.freq = 50, random.order = F, scale = c(3, 0.3), colors = colors, us

#WordCloud for Trump's Speech
colors = brewer.pal(6, "Dark2")
wordcloud(wf.b$word, wf.b$freq, min.freq = 50, random.order = F, scale = c(5, 0.3), colors = colors, us
```




We could see that there are some words that both used a lot by Trump and Hillary such as **people, going, will**. That make sense because these words represent good expectation. In addition to this, there is something makes me feel interested:

1. Word 'Donald' has a high frequency in Hillary's speech whereas word 'Hillary' has a high frequency in Trump's speech. I bet that is because they always humiliate each other after calling the enemies' names.
2. The size of the word represents its frequency. So we could see that the size of some words are super big in Trump's wordcloud and others are much small. That means Trump tends to use parallelism or use words repeatedly.

Specifically, let us see the first sentence in Trump's victory speech:

**"Thank you. Thank you very much, everyone. Sorry to keep you waiting. Complicated business, complicated. Thank you very much"**

Then let us see what Obama said in his victory speech:

**"If there is anyone out there who still doubts that America is a place where all things are possible, who still wonders if the dream of our founders is alive in our time, who still questions the power of our democracy, tonight is your answer"**

So, by comparison, I don't think Trump's sentence uses parallelism. Trump just says some words repeatedly.

But Mr.Trump doesn't agree with the viewpoint that he likes to speak words repeatedly, here is his contradictions:


Thank you Trump, you just perfectly proved my argument.

### Generating Keyness

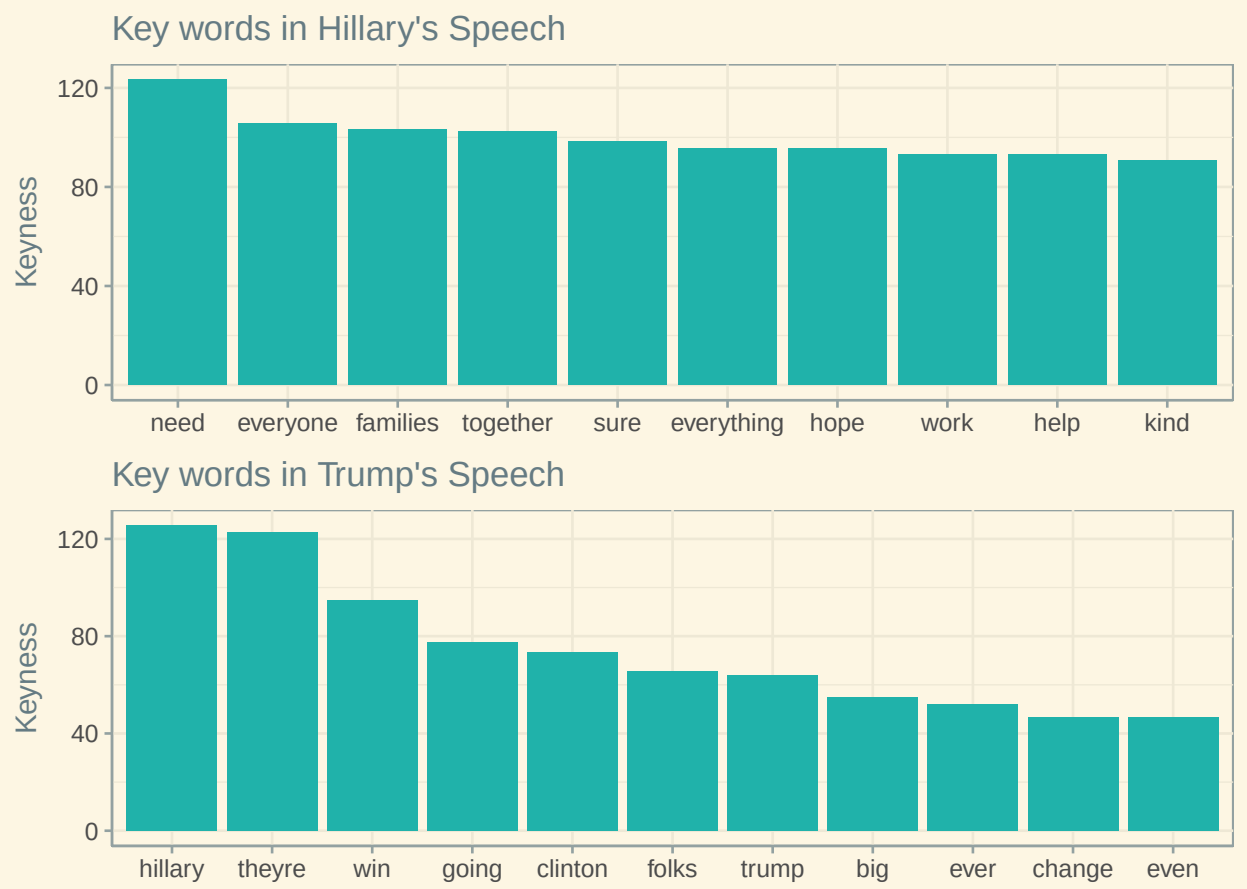
After seeing the wordcloud, I would like to use another method to find the difference between their speeches. So here comes keyness.

Keyness is a common calculation in corpus linguistics that compares word frequencies in one corpus (a target corpus which is Trump's Corpus here) to another (a reference corpus which is Clinton's Corpus here). Here we use Log-likelihood to quantify the keyness:

Log-likelihood tells us how much evidence we have for a difference between two corpora. Here is the link for detailed explanation: Log-likelihood explanation

```
wf = merge(wf.a,wf.b,by="word", all=TRUE)
wf[is.na(wf)] = 0

#calculate totals and normalized frequencies
total.a = sum(wf.a$freq)
total.b = sum(wf.b$freq)

normalize.a = function (frequency.a) {
  normal.a = (frequency.a/total.a)*100
  return(normal.a)
}

percent.x = mapply(normalize.a, wf$freq.x)
wf$percent.x = percent.x
```

```r
normalize.b = function (frequency.b) {
  normal.b = (frequency.b/total.b)*100
  return(normal.b)
}

percent.y = mapply(normalize.b, wf$freq.y)
wf$percent.y = percent.y

#calculate keyness
log.like = function(frequency.a, frequency.b) {
  expected.a = total.a*((frequency.a+frequency.b)/(total.a+total.b))
  expected.b = total.b*((frequency.a+frequency.b)/(total.a+total.b))
  L1 = if(frequency.a == 0) 0 else (frequency.a*log(frequency.a/expected.a))
  L2 = if(frequency.b == 0) 0 else (frequency.b*log(frequency.b/expected.b))
  likelihood = 2*(L1 + L2)
  return(likelihood)
}

keyness = mapply(log.like, wf$freq.x, wf$freq.y)
wf$keyness = keyness

compare = function (x,y) {
  if(x > y) return("x>y")
  if(x < y) return("x<y")
  if (x==y) return ("x=y")
}

x.vs.y = mapply(compare, wf$percent.x, wf$percent.y)
wf$x.vs.y = x.vs.y

wf = wf[with(wf, order(-keyness)), ]


#Visualization for the differences between these two corpus
clinton_part = wf %>% filter(x.vs.y == 'x>y') %>% top_n(10, wt = keyness)
clinton_part$word = factor(clinton_part$word, levels = clinton_part$word[order(clinton_part$keyness, de
p1 = ggplot(data = clinton_part, aes(word, keyness)) +
  geom_col(fill = 'lightseagreen') +
  labs(x = NULL, y = 'Keyness', title = "Key words in Hillary's Speech") +
  theme_solarized(light = T)


trump_part = wf %>% filter(x.vs.y == 'x<y') %>% top_n(10, wt = keyness)
trump_part$word = factor(trump_part$word, levels = trump_part$word[order(trump_part$keyness, decreasing
p2 = ggplot(data = trump_part, aes(word, keyness)) +
  geom_col(fill = 'lightseagreen') +
  labs(x = NULL, y = 'Keyness', title = "Key words in Trump's Speech") +
  theme_solarized(light = T)

grid.arrange(p1,p2, nrow=2)
```




We could see that by comparison, Hillary tends to use more positive words such like 'everyone/together/work/families'. Whereas Trump speaks more of Hillary's names. I could imagine how hard Trump has tried to defame Hillary.

**Readability Comparison**

Then I perform readability tests that designed to indicate how difficult a passage in English is to understand. Here I use Flesch - Kincaid readability tests to compare their speeches. This test will give us two results: The U.S. Grade level and the age level of the writers.

Here is the basic formula of F-K test:

$$0.39(\frac{total\ words}{total\ sentences}) + 11.8(\frac{total\ syllables}{total\ words}) - 15.59$$

For more detailed information of Flesch - Kincaid test, please refer to this link:

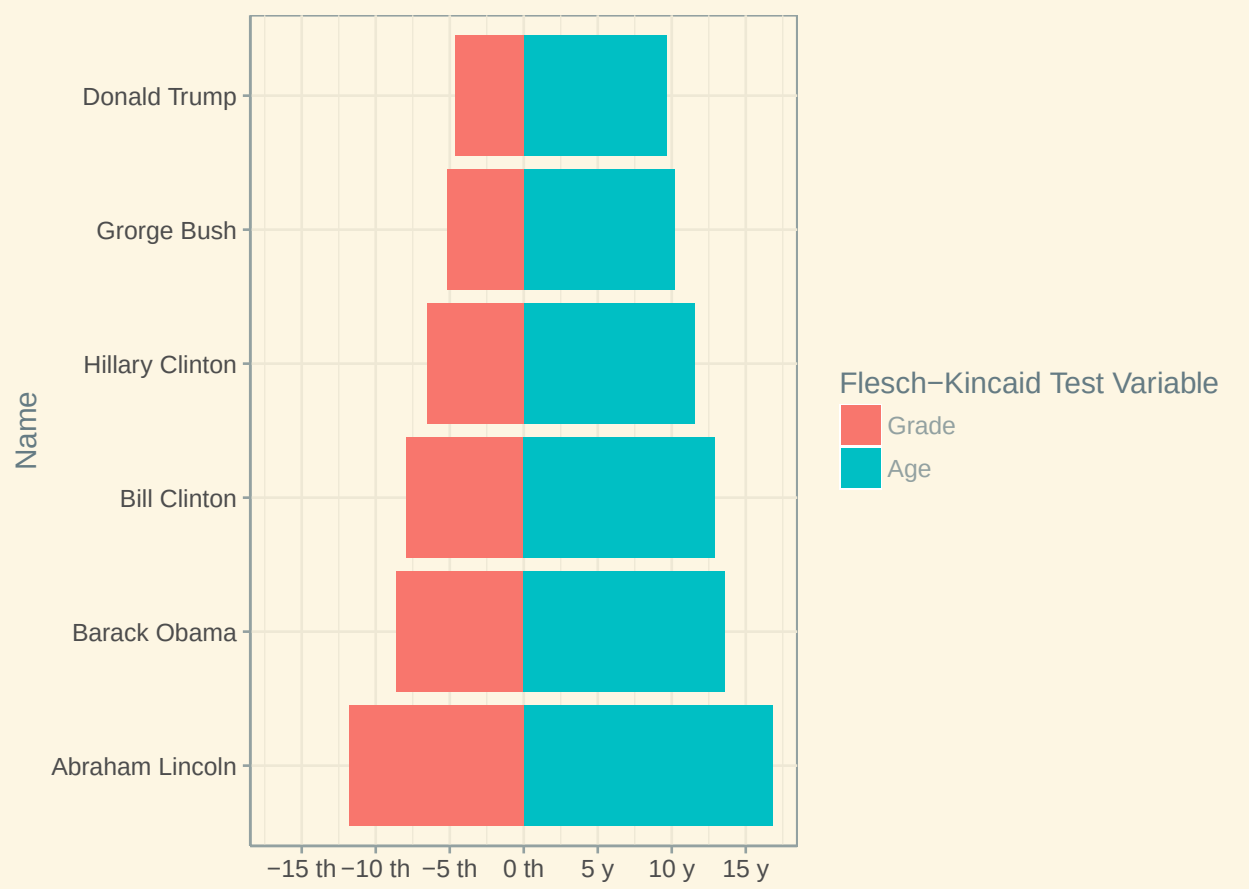
Flesch - Kincaid readability tests

Fortunately, the package 'koRpus' has the function to test the readability. So here I use the `readability` function from `KoRpus` package.

This time, I don't just compare the readability within Hillary and Trump, I also include the latest presidents 'Barack Obama', 'George Bush', 'Bill Clinton' and one old president 'Abraham Lincoln'. Let us see the result.

```
#Compare the readability between Clinton and Trump
##Tokenize the target text
clinton_tokenize = tokenize('../data/Clinton-Trump Corpus/Clinton', lang = 'en')
```

```
trump_tokenize = tokenize('../data/Clinton-Trump Corpus/Trump', lang = 'en')

##Calculate the Readability
clinton.readability = koRpus::readability(clinton_tokenize, index = 'Flesch.Kincaid', quiet = T)
trump.readability = koRpus::readability(trump_tokenize, index = 'Flesch.Kincaid', quiet = T)


##Tokenize five presidents's text
obama_tokenize = tokenize('../data/InauguralSpeeches/Barack Obama', lang = 'en')
bush_tokenize = tokenize('../data/InauguralSpeeches/George Bush', lang = 'en')
bill_clinton_tokenize = tokenize('../data/InauguralSpeeches/Bill Clinton', lang = 'en')
lincoln_tokenize = tokenize('../data/InauguralSpeeches/Abraham Lincoln', lang = 'en')
##Calculate the readability
obama.readability = koRpus::readability(obama_tokenize, index = 'Flesch.Kincaid', quiet = T)
bush.readability = koRpus::readability(bush_tokenize, index = 'Flesch.Kincaid', quiet = T)
bill_clinton.readability = koRpus::readability(bill_clinton_tokenize, index = 'Flesch.Kincaid', quiet =
lincoln.readability = koRpus::readability(lincoln_tokenize, index = 'Flesch.Kincaid', quiet = T)

#Generate a data frame to store all these information
df_readability = data.frame(
  Name = c('Hillary Clinton', 'Donald Trump', 'Barack Obama', 'Grorge Bush', 'Bill Clinton', 'Abraham Li
  Grade = c(clinton.readability@Flesch.Kincaid$grade, trump.readability@Flesch.Kincaid$grade, obama.read
  Age = c(clinton.readability@Flesch.Kincaid$age, trump.readability@Flesch.Kincaid$age, obama.readabilit

#visualization
##sort the data frame
df_readability$Name = factor(df_readability$Name, levels = df_readability$Name[order(df_readability$Age

df_r = df_readability %>% melt(id.vars = 'Name')
ggplot(df_r, aes(x = Name, fill = variable, y = ifelse(test = variable == "Grade", yes = -value, no = va
  geom_bar(stat = "identity") +
  scale_y_continuous(labels = c(paste(seq(-20,0,5), 'th'), paste(seq(5,20,5), 'y')), limits = max(df_r$v
  labs(x = 'Name', y = NULL) +
  scale_fill_discrete('Flesch-Kincaid Test Variable') +
  coord_flip() +
  theme_solarized(light = T)
```




Let us look at this pyramid plot to see the differences between their speeches' readability.

No surprise. Mr.President speaks like a fourth grade boy who is about nine years old. That is very interesting, So here are some sentences that I find from his speech:

### "I'm really rich"

### "We're going to do a wall; we're going to have a big, fat beautiful door on the wall"

### "The system is a beatiful system when it works"

I think Trump can be a good primary school English teacher.

---

## Part Two : Sentence Analysis - Sentiment Analysis

After having analyzed the word similarities and differences between their speeches, I then start to perform sentence analysis. Here I will use sentences as units to perform sentiment analysis.

For each extracted sentence, I apply two sentiment analysis methods:

- NRC Sentiment Lexion. Here is the link for detailed explaination: NRC sentiment lexion The NRC Emotion Lexicon is a list of English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive). The annotations were manually done by crowdsourcing.
- syuzhet Sentiment Score. Here is the link for detailed explaination: syuzhet Sentiment Score

| Example Text | Sentiment Score |
|---|---|
| good | 3 |
| bad | -3 |
| mediocre | 0 |
| amazing | 4 |
| good and bad | 0 |
| amazing and bad | 1 |

Also, I assign an sequential id to each sentence in a speech (`sent.id`) and also calculated the number of words in each sentence as *sentence length* (`word.count`).

### Data Pre-processing

Load the data and put all information into a dataframe.

```
#Load The File
setwd('../data/Clinton-Trump Corpus')
clinton1 = paste(readLines("Clinton/1.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton2 = paste(readLines("Clinton/2.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton3 = paste(readLines("Clinton/3.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton4 = paste(readLines("Clinton/4.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton5 = paste(readLines("Clinton/5.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton6 = paste(readLines("Clinton/6.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton7 = paste(readLines("Clinton/7.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton8 = paste(readLines("Clinton/8.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton9 = paste(readLines("Clinton/9.txt", n=-1, skipNul=TRUE), collapse=" ")
clinton10 = paste(readLines("Clinton/10.txt", n=-1, skipNul=TRUE), collapse=" ")

trump1 = paste(readLines("Trump/1.txt", n=-1, skipNul=TRUE), collapse=" ")
trump2 = paste(readLines("Trump/2.txt", n=-1, skipNul=TRUE), collapse=" ")
trump3 = paste(readLines("Trump/3.txt", n=-1, skipNul=TRUE), collapse=" ")
trump4 = paste(readLines("Trump/4.txt", n=-1, skipNul=TRUE), collapse=" ")
trump5 = paste(readLines("Trump/5.txt", n=-1, skipNul=TRUE), collapse=" ")
trump6 = paste(readLines("Trump/6.txt", n=-1, skipNul=TRUE), collapse=" ")
trump7 = paste(readLines("Trump/7.txt", n=-1, skipNul=TRUE), collapse=" ")
trump8 = paste(readLines("Trump/8.txt", n=-1, skipNul=TRUE), collapse=" ")
trump9 = paste(readLines("Trump/9.txt", n=-1, skipNul=TRUE), collapse=" ")
trump10 = paste(readLines("Trump/10.txt", n=-1, skipNul=TRUE), collapse=" ")

#Hillary Speeches Data Frame
clinton.speeches=data.frame(
  Name = rep ("Hillary Clinton", 10),
  File = paste0('clinton', 1:10),
  Words=c(word_count(clinton1),word_count(clinton2),word_count(clinton3),word_count(clinton4),word_coun
  fulltext=c(clinton1,clinton2,clinton3,clinton4,clinton5,clinton6,clinton7,clinton8,clinton9,clinton10
)

#Trump Speeches Data Frame
trump.speeches=data.frame(
  Name = rep ("Donald Trump", 10),
  File = paste0('trump', 1:10),
  Words=c(word_count(trump1),word_count(trump2),word_count(trump3),word_count(trump4),word_count(trump5
  fulltext=c(trump1,trump2,trump3,trump4,trump5,trump6,trump7,trump8,trump9,trump10)
)

#Complete Data Frame
speech.list=rbind(clinton.speeches, trump.speeches)
```

Here is what speech.list data frame looks like:

| Name | File | Words | fulltext |
|---|---|---|---|
| Hillary Clinton | clinton1 | 3625 | Hello Dade City. Wow. I am so excited... |
| Hillary Clinton | clinton2 | 1941 | Wow! Thank you so much! And thank you for being... |
| Hillary Clinton | clinton3 | 3676 | t is great to be here. Thank you. I – I am very happy... |
| Hillary Clinton | clinton4 | 3996 | I – I am – I am so, so touched and so grateful to be... |

### Create the sentiment dataframe

Using the data frame created before, extract the sentence and perform sentiment analysis by the two methods I just mentioned.

```
#Two methods to quantify the sentiment of each sentence (get_sentiment/get_nrc_sentiment)

sentence.list = NULL

for(i in 1:nrow(speech.list)){
  sentences = sent_detect(speech.list$fulltext[i],
                          endmarks = c("?", ".", "!", "|",";"))
  if(length(sentences)>0){
    emotions = get_nrc_sentiment(sentences) #NRC sentiment analysis
    sentiment_score = as.numeric(get_sentiment(sentences, method = 'syuzhet')) #syuzhet sentiment score
    word.count = word_count(sentences)
    emotions = diag(1/(word.count+0.01))%*%as.matrix(emotions)
    sentence.list = rbind(sentence.list,
                          cbind(speech.list[i,-ncol(speech.list)],
                                sentences = as.character(sentences),
                                word.count,
                                sentiment_score = sentiment_score,
                                emotions,
                                sent.id = 1:length(sentences)
                                ))

  }
}

#Some non-sentences exist in raw data due to erroneous extra end-of sentence marks
sentence.list = sentence.list %>% filter(!is.na(word.count))
```

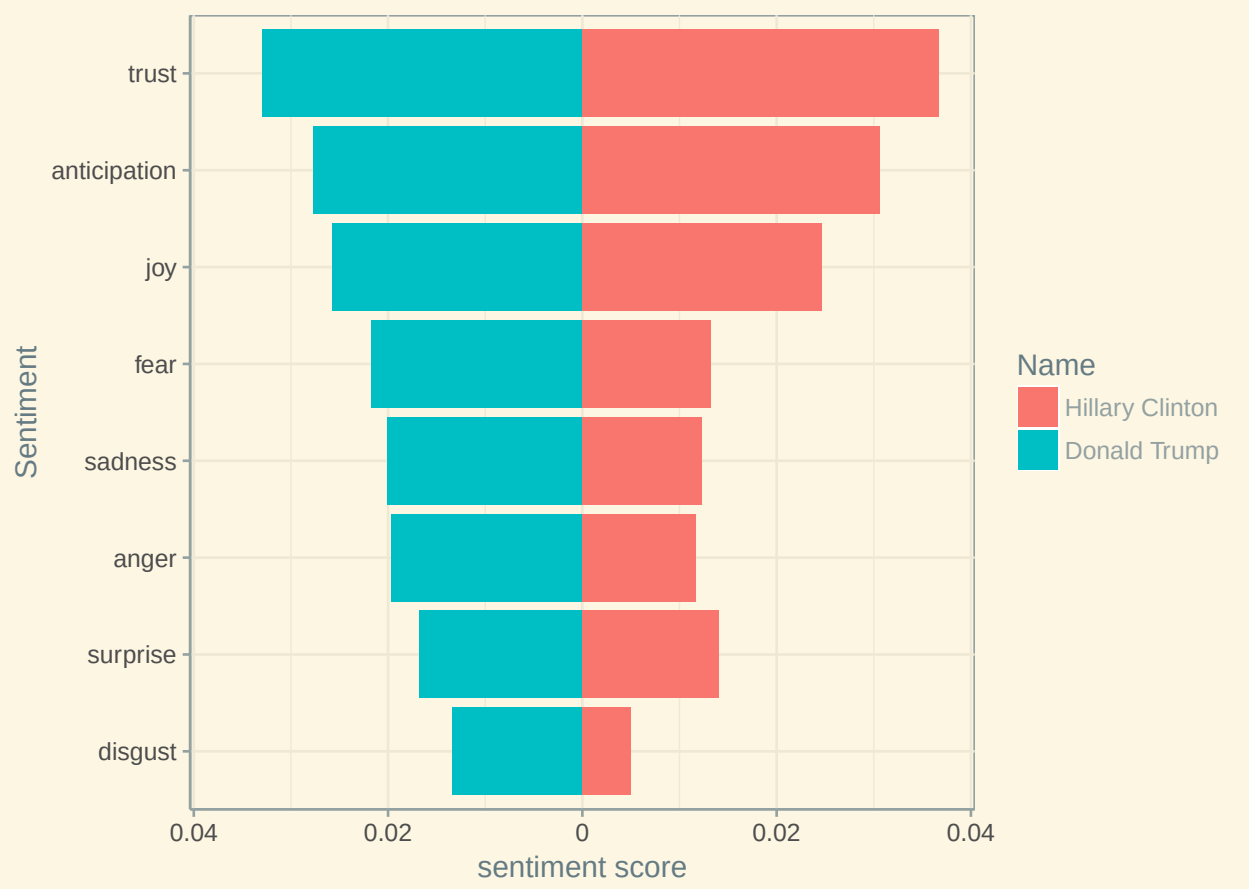
Here is what sentence.list data frame looks like:

| Name | File | Words | sentence | word.count | sentiment_score | anger | anticipation | disgust | fear | joy | sadness | surprise | trust | negative | positive | sent.id |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hillary Clinton | hillary | 3625 | Hello Dade City. | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Hillary Clinton | hillary | 3625 | Wow. | 1 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| Hillary Clinton | hillary | 3625 | I am so excited... | 16 | 3 | 0 | 0.06 | 0 | 0 | 0.12 | 0 | 0.12 | 0.12 | 0 | 0.12 | 3 |

**Sentiment Comparison**

- Create a summary table contains the mean NRC score and also the mean syuzhet score
- visualization

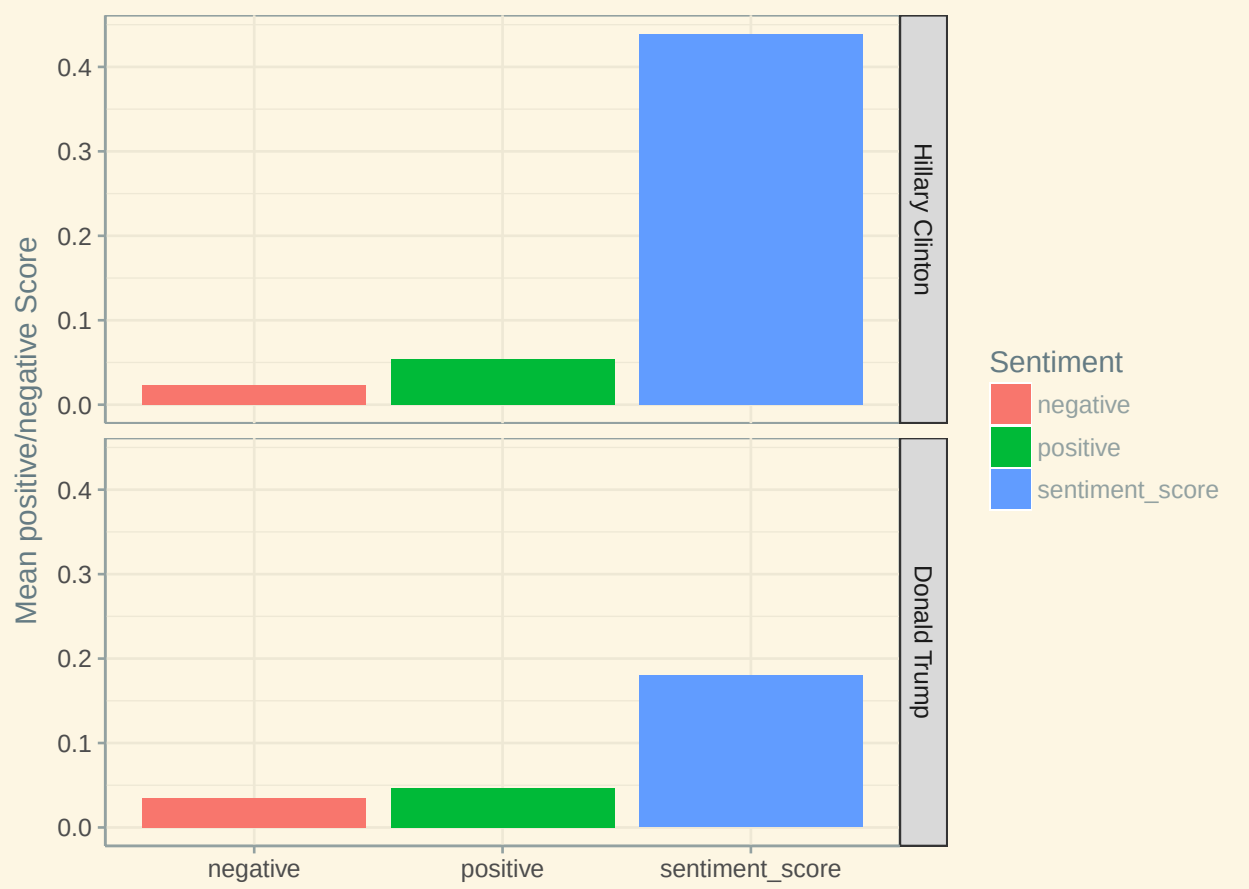
```
#Create a summary table for sentiment analysis
sentiment.summary = tbl_df(sentence.list) %>%
  group_by(Name) %>%
  summarise(
    anger=mean(anger),
    anticipation=mean(anticipation),
    disgust=mean(disgust),
    fear=mean(fear),
    joy=mean(joy),
    sadness=mean(sadness),
    surprise=mean(surprise),
    trust=mean(trust),
    negative=mean(negative),
    positive=mean(positive),
    sentiment_score = mean(sentiment_score)
  )


df_t = sentiment.summary %>% select(-c(negative, positive, sentiment_score)) %>% melt(id.vars = 'Name')
ggplot(df_t, aes(x = reorder(variable, value), y = ifelse(test = Name == "Donald Trump", yes = -value, r
  geom_bar(stat = 'identity') +
  scale_y_continuous(labels = abs, limits = max(df_t$value) * c(-1,1))+
  coord_flip() +
  labs(x = 'Sentiment', y = 'sentiment score') +
  theme_solarized(light = T)
```




Based on these comparison, we could see that both of them talked a lot about their confidence in wining this presidency. The difference is that Donald Trump is more emotional. We see that the propotion of sentences related to fear, sadness and anger are much more than Hillary' speeches, which also means that Trump speaks less positive than Hillary's.

The plot of the syuzhet sentiment score could futher prove this:

```
sentiment.summary %>% select(c(Name, negative, positive, sentiment_score)) %>% melt(id.vars = 'Name') %
  geom_col() +
  labs(x = NULL, y = 'Mean positive/negative Score') +
  scale_fill_discrete('Sentiment')+
  facet_grid(Name~.) +
  theme_solarized(light = T)
```




**The sentence related to that sentiment**

```
## [1] "The crime is horrible."
## [2] "God bless you, everybody."
## [3] "Bad, I don't want to lose Texas."
## [4] "We will stop illegal immigration, deport every last criminal alien, and dismantle every criminal
## [5] "God bless you, everybody."
## [6] "Cancel every illegal Obama executive order."
## [7] "Good luck with that negotiation."
## [8] "That's a pretty good statement."
```

---

# How Did Donald Trump Win the Presidency?

**My conclusion is :**

**Speak Like a Emotional Nine Years Old Boy!**

**Thank You Very Much!**