Text classification - Speeches by Obama and Trump

R Cafe - Jonathan - j.debruin1@uu.nl

11/25/2019

Text Classification

Text classification is the process of assigning text into organized groups.

Applications

- Customer service
- Language Detection
- Spam filters
- ...

Text Mining with R

Text mining in R is challenging (without external tools).

> *We developed the tidytext R package because we were familiar with many methods for data wrangling and visualization, but couldn't easily apply these same methods to text. (Silge and Robinson 2016)*

- Text mining package `tidytext`
- Book "Text mining with R (Silge and Robinson, 2016)"
- www.tidytextmining.com/

Text Mining with R

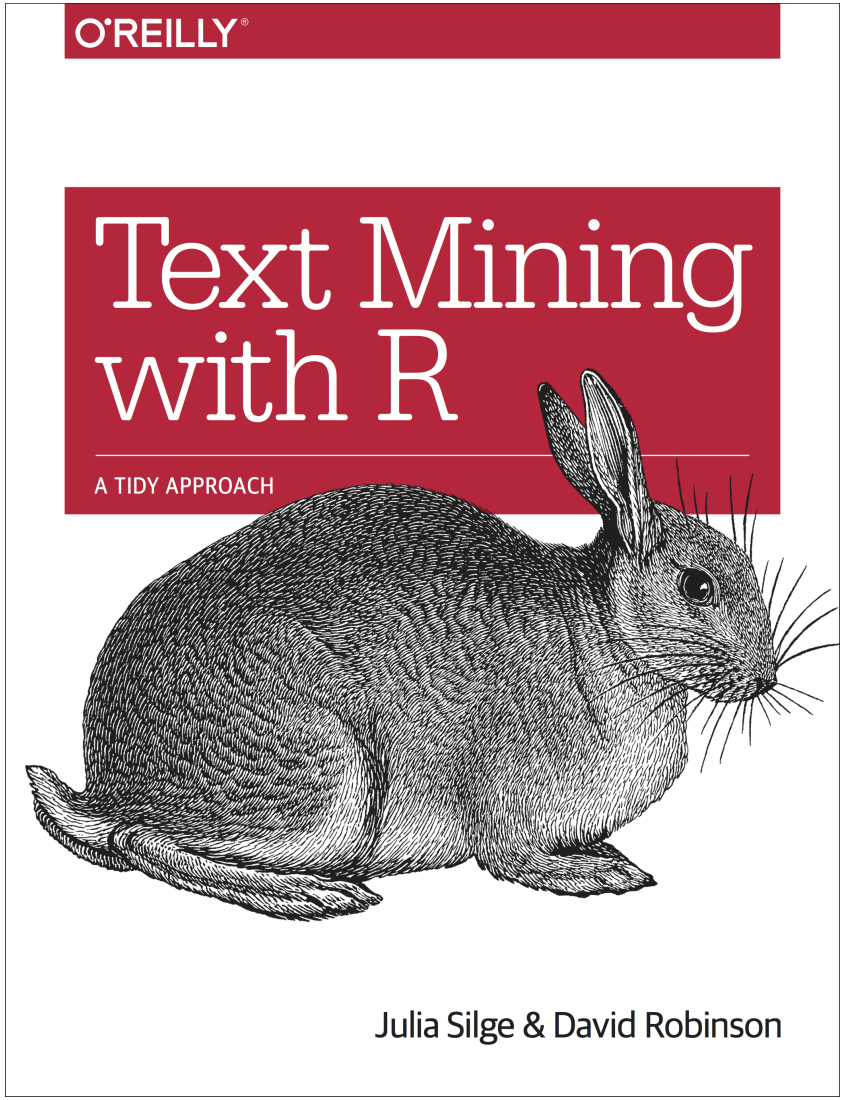


Figure 1:

Recap: Tidy data

Tidy data has a specific structure (Wickham 2014):

- Each variable is a column
- Each observation is a row
- Each type of observational unit is a table

Recap: Non-tidy data (iris)

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1           5.1         3.5          1.4         0.2  setosa
## 2           4.9         3.0          1.4         0.2  setosa
## 3           4.7         3.2          1.3         0.2  setosa
## 4           4.6         3.1          1.5         0.2  setosa
## 5           5.0         3.6          1.4         0.2  setosa
## 6           5.4         3.9          1.7         0.4  setosa
## 7           4.6         3.4          1.4         0.3  setosa
## 8           5.0         3.4          1.5         0.2  setosa
## 9           4.4         2.9          1.4         0.2  setosa
## 10          4.9         3.1          1.5         0.1  setosa
```

Recap: Tidy data (iris)

```
##     id Species      measure value
## 1    1  setosa  Sepal.Width   3.5
## 2    1  setosa Sepal.Length   5.1
## 3    1  setosa  Petal.Width   0.2
## 4    1  setosa Petal.Length   1.4
## 5    2  setosa  Sepal.Width   3.0
## 6    2  setosa Sepal.Length   4.9
## 7    2  setosa  Petal.Width   0.2
## 8    2  setosa Petal.Length   1.4
## 9    3  setosa  Sepal.Width   3.2
## 10   3  setosa Sepal.Length   4.7
```

Tidy text

Definition: tidy text format is a table with ***one-token-per-row***

- A token is a meaningful unit of text, such as a word, that we are interested in using for analysis, and tokenization is the process of splitting text into tokens.

Packages for text classification

```
# default tidyverse packages
library(tidyverse)
library(lubridate)

# text mining related
library(tidytext)
library(tm)
library(textdata)
library(wordcloud)

# machine learning
library(caret)
library(randomForest)
```

The President's Weekly Address

- Example: https://www.presidency.ucsb.edu/documents/the-presidents-weekly-address-431

My fellow Americans, the heartbreaking devastation and suffering caused by Hurricane Harvey has profoundly affected our entire Nation. Many homes and communities have been destroyed, many lives have been upended, and tragically, some have lost their lives in this catastrophic storm. We pray for the victims and their families and all of those who have been displaced from their homes.

At this very moment, heroic efforts continue to keep safe those threatened by this natural disaster. I want to say a special word of thanks to our amazing first responders: our police and law enforcement officers, firefighters, Coast Guard, National Guard, EMS, doctors, nurses, hospital workers, and volunteers who have traveled from all across the country. Thousands of people have come together to prevent loss of life and ensure safety, and we are incredibly grateful for their courage, their professionalism, and their sacrifice. They are an inspiration to all of us.

The President's Weekly Address - read data

- Run `data_downloader.R`

```
president_speeches <- read_csv(
  "data/speeches.csv",
  col_types = cols(name="f")
) %>%
  mutate(date=mdy(date))
```

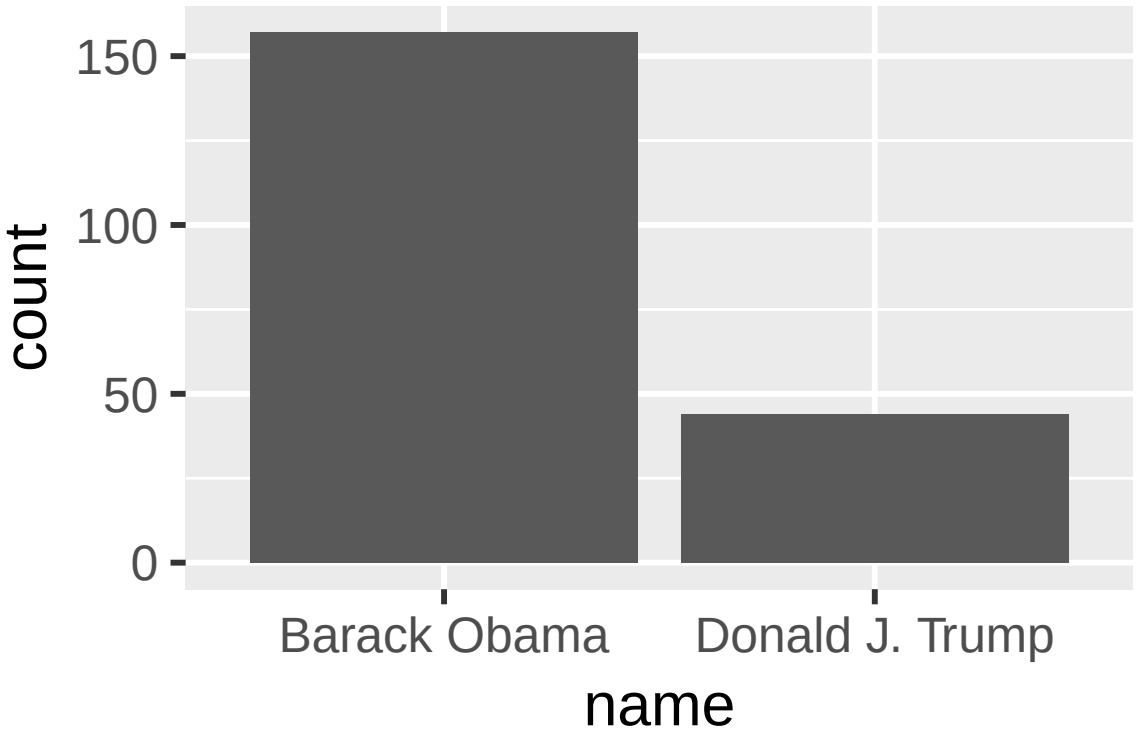
The President's Weekly Address - preview

```
head(president_speeches)
```

```
## # A tibble: 6 x 4
##      id date       name       speech
##   <dbl> <date>     <fct>      <chr>
## 1   251 2015-03-14 Barack Ob~ "\n    Hi, everybody. Earlier this week, I v~
## 2   252 2015-03-21 Barack Ob~ "\n    Hi, everybody. One of the most import~
## 3   253 2015-04-04 Barack Ob~ "\n    This week, together with our allies a~
## 4   254 2015-03-28 Barack Ob~ "\n    Hi, everybody. Five years ago, after ~
## 5   255 2015-03-07 Barack Ob~ "\n    Hi, everybody. Sunday is Internationa~
## 6   256 2015-02-28 Barack Ob~ "\n    Hi, everybody. In America, we believe~
```

The President's Weekly Address

```
president_speeches %>%
  ggplot() +
  geom_histogram(aes(name), stat="count")
```



The President's Weekly Address - tokenizing & tidy text

```
(president_tokens <- unnest_tokens(president_speeches, word, speech))
```

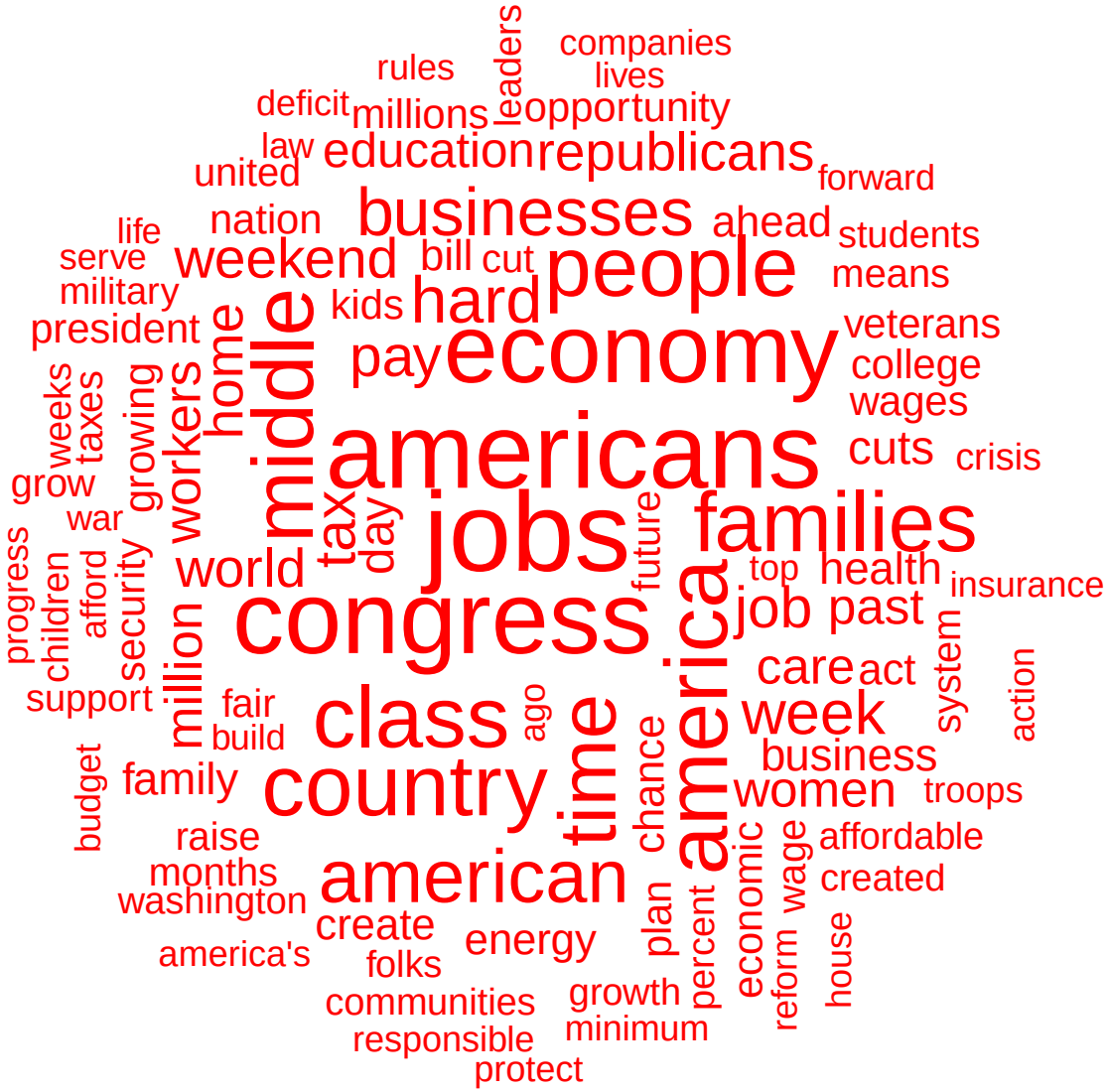
```
## # A tibble: 110,164 x 4
##       id date       name          word
##    <dbl> <date>     <fct>         <chr>
##  1   251 2015-03-14 Barack Obama hi
##  2   251 2015-03-14 Barack Obama everybody
##  3   251 2015-03-14 Barack Obama earlier
##  4   251 2015-03-14 Barack Obama this
##  5   251 2015-03-14 Barack Obama week
##  6   251 2015-03-14 Barack Obama i
##  7   251 2015-03-14 Barack Obama visited
##  8   251 2015-03-14 Barack Obama with
##  9   251 2015-03-14 Barack Obama students
## 10   251 2015-03-14 Barack Obama at
## # ... with 110,154 more rows
```

The President's Weekly Address - Word cloud

```
library(wordcloud)

president_tokens %>%
  filter(name=="Donald J. Trump") %>%
  anti_join(stop_words, by="word") %>%
  count(word, sort = TRUE) %>%
  with(
    wordcloud(
      word, n, scale=c(5, 1),
      max.words = 100,
      random.order=FALSE,
      rot.per=0.3,
      colors='blue')
  )
```

The President's Weekly Address - Word cloud Trump

The President's Weekly Address - Word cloud Obama



Term frequency

- **Term Frequency (tf)** - A measure of how important a word is.
- *How frequently a word occurs in a document. There are words in a document, however, that occur many times but may not be important; in English, these are probably words like "the", "is", "of", and so forth. We might take the approach of adding words like these to a list of stop words and removing them before analysis, but it is possible that some of these words might be more important in some documents than others. (Text Mining with R, 2019)*

The President's Weekly Address - Term frequency

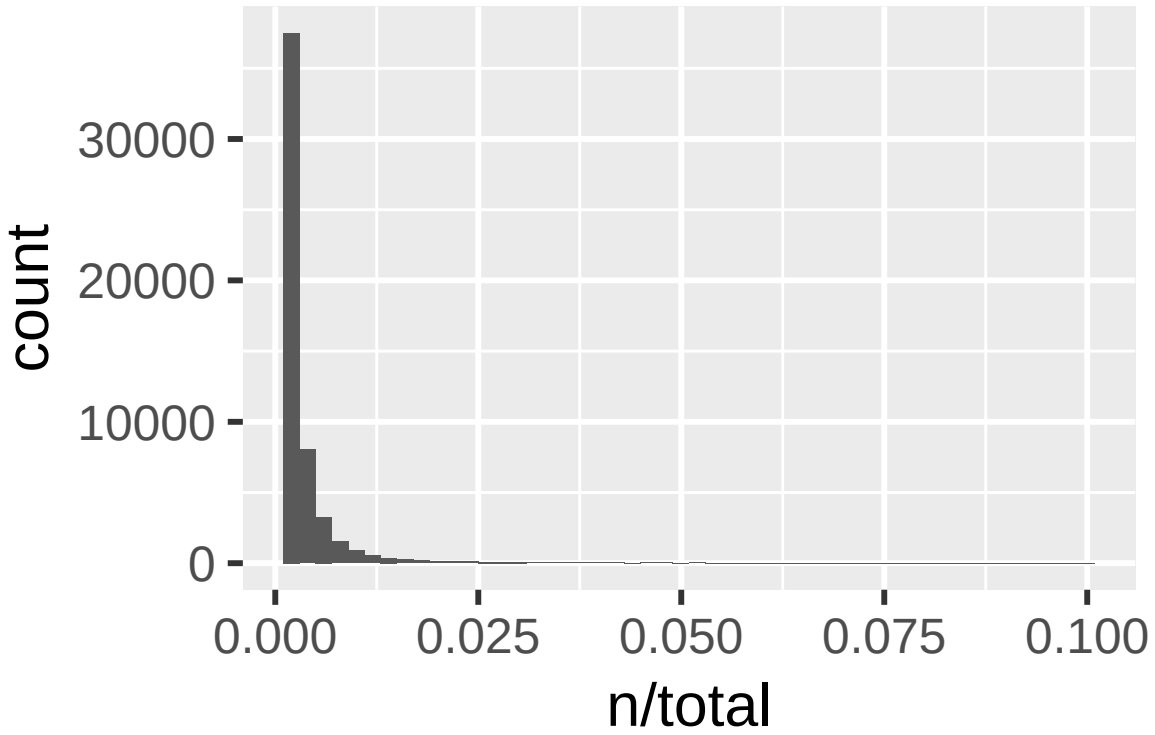
```
president_words <- president_tokens %>% count(id, name, word, sort=TRUE)

total_words <- president_words %>%
  group_by(id) %>%
  summarize(total = sum(n))

(president_words <- left_join(president_words, total_words, by = "id"))
```

```
## # A tibble: 53,812 x 5
##       id name             word      n total
##    <dbl> <fct>            <chr> <int> <int>
##  1   324 Barack Obama     the      52   958
##  2   293 Barack Obama     the      51   984
##  3   293 Barack Obama     and      49   984
##  4   273 Barack Obama     the      46   752
##  5   432 Donald J. Trump and      45   770
##  6   310 Barack Obama     the      44   668
##  7   416 Donald J. Trump the      44   629
##  8   326 Barack Obama     the      43   777
##  9   348 Barack Obama     the      40   816
## 10   374 Barack Obama     to       39   603
## # ... with 53,802 more rows
```

Term frequency - Zipf's distribution

- Long-tailed distribution (Zipf's)

```
ggplot(president_words, aes(n/total)) +
  geom_histogram(bins=50)
```



Inverse Document Frequency

- **Inverse Document Frequency (IDF)** - IDF decreases the weight for commonly used words and increases the weight for words that are not used very much in a collection of documents.
- *The statistic **tf-idf** is intended to measure how important a word is to a document in a collection (or corpus) of documents, for example, to one novel in a collection of novels or to one website in a collection of websites. (Text Mining with R, 2019)*

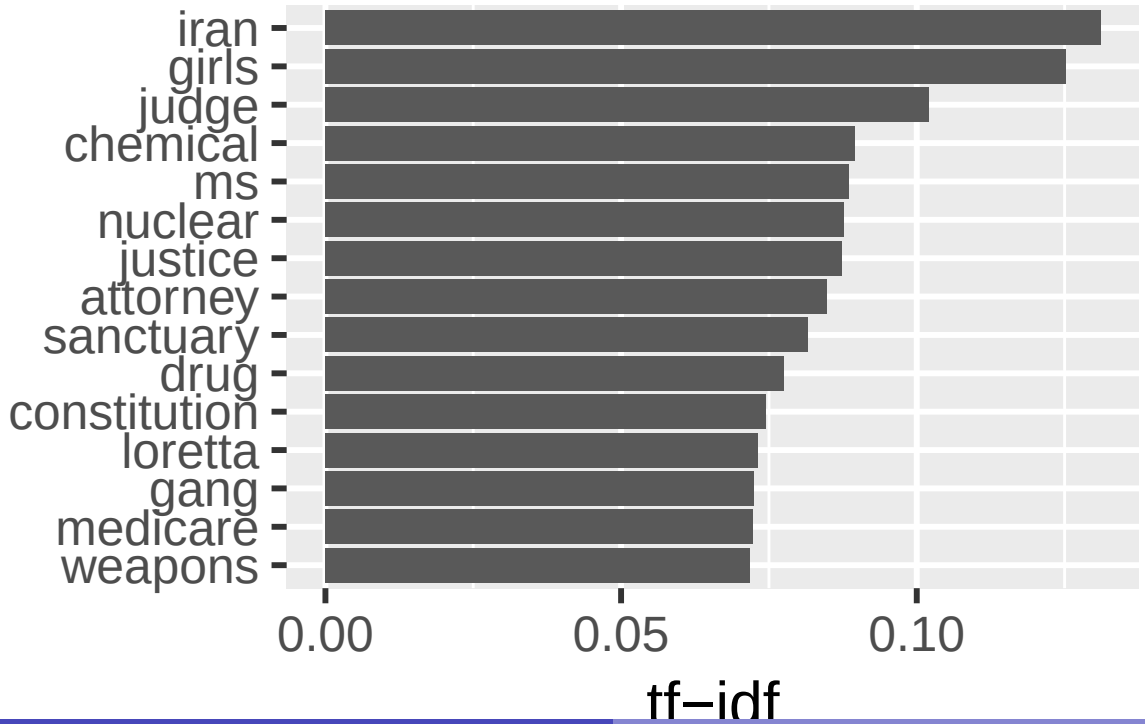
The President's Weekly Address - Inverse Document Frequency

```
(president_words <- president_words %>%
  bind_tf_idf(word, id, n))
```

```
## # A tibble: 53,812 x 8
##       id name            word      n total     tf   idf tf_idf
##    <dbl> <fct>           <chr> <int> <int>  <dbl> <dbl>  <dbl>
##  1   324 Barack Obama    the      52   958 0.0543     0      0
##  2   293 Barack Obama    the      51   984 0.0518     0      0
##  3   293 Barack Obama    and      49   984 0.0498     0      0
##  4   273 Barack Obama    the      46   752 0.0612     0      0
##  5   432 Donald J. Trump and      45   770 0.0584     0      0
##  6   310 Barack Obama    the      44   668 0.0659     0      0
##  7   416 Donald J. Trump the      44   629 0.0700     0      0
##  8   326 Barack Obama    the      43   777 0.0553     0      0
##  9   348 Barack Obama    the      40   816 0.0490     0      0
## 10   374 Barack Obama    to       39   603 0.0647     0      0
## # ... with 53,802 more rows
```

The President's Weekly Address - TF-IDF

```
president_words %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  top_n(15) %>%
  ggplot(aes(word, tf_idf)) +
    geom_col() +
    labs(x = NULL, y = "tf-idf") +
    coord_flip()
```



from tidy to DocumentTermMatrix

Tidy text format is not suitable for machine learning.

```
library(tm)

(president_tfidf = president_words %>%

  arrange(id) %>%

  # cast to document term matrix
  cast_dtm(id, word, tf_idf) %>%

  # remove sparse terms
  removeSparseTerms(0.9))
```

```
## <<DocumentTermMatrix (documents: 201, terms: 552)>>
## Non-/sparse entries: 33732/77220
## Sparsity           : 70%
## Maximal term length: 14
## Weighting          : term frequency (tf)
```

Split dataset into train and test set

```
library(caret)

set.seed(535)

trainIndex <- createDataPartition(president_speeches$name, p = .6,
                                  list = FALSE,
                                  times = 1)

presidentTrain <- president_tfidf[ trainIndex,]
presidentTest  <- president_tfidf[-trainIndex,]

presidentTrain
```

```
## <<DocumentTermMatrix (documents: 122, terms: 552)>>
## Non-/sparse entries: 20502/46842
## Sparsity           : 70%
## Maximal term length: 14
## Weighting          : term frequency (tf)
```

Machine Learning - Random Forest

- Fit Random Forest on the train data

```
library(randomForest)

(classifier <- randomForest(
  x = as.data.frame(as.matrix(president_tfidf[trainIndex,])),
  y = as.factor(president_speeches[trainIndex,]$name),
  nTree = 10))
```

```
##
## Call:
##  randomForest(x = as.data.frame(as.matrix(president_tfidf[trainIndex,      ])),
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 23
##
##         OOB estimate of  error rate: 7.38%
## Confusion matrix:
##                 Barack Obama Donald J. Trump class.error
## Barack Obama              94               1  0.01052632
## Donald J. Trump            8              19  0.29629630
```

Machine Learning - Random Forest

- Fit Random Forest on the train data

```
(president_pred <- predict(
  classifier,
  newdata = as.data.frame(as.matrix(president_tfidf[-trainIndex,]))
))
```

```
##              252              254              255              258
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
##              262              263              266              269
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
##              270              271              272              274
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
##              276              278              279              280
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
##              282              283              284              287
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
##              295              303              305              307
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
##              311              312              319              320
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
##              321              322              331              335
##    Barack Obama     Barack Obama     Barack Obama     Barack Obama
```

Validation - Confusion matrix

- https://en.wikipedia.org/wiki/Confusion_matrix

```
(cm <- table(
  president_speeches[-trainIndex,]$name,
  president_pred
))
```

```
##                     president_pred
##                      Barack Obama Donald J. Trump
##    Barack Obama                62                0
##    Donald J. Trump              2               15
```

Questions?

Thanks for attending.

R Cafe 15:00 - 17:00