# Text Mining and Modelling in R

Tom Merritt Smith
@tmrtsmith

SatRday April 6th, 2019

# Text mining and modelling in R

```
library(r2py)
```

# Text mining and modelling in R

- Text Mining (Exploratory analysis)
- Topic Modelling and Word Embeddings (Unsupervised techniques)
- Text Classification (Supervised techniques)
- Neural Network musings (witchcraft!)

# Text Mining

- Counting
- Weighing
- Plotting
- Not Wordclouds

# Counting

```
library(tidytext)
trump_tweets %>%
  # split a column into tokens
  # using the tokenizers package
  unnest_tokens(word, text) %>%
  # remove stop_words
  anti_join(stop_words) %>%
  # this is pretty pythonic, tbh
  count(word, sort=TRUE) %>%
  head()
```

# Counting

```
## Joining, by = "word"
```

```
## # A tibble: 6 x 2
##   word           n
##   <chr>      <int>
## 1 amp          568
## 2 people       409
## 3 president    371
## 4 border       356
## 5 trump        330
## 6 country      310
```

# Counting

```
trump_tweets %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2) %>%
  separate(bigram, c("word1", "word2"), sep = " ") %>%
  filter(!word1 %in% stop_words$word,
         !word2 %in% stop_words$word) %>%
  count(word1, word2, sort=TRUE) %>%
  slice(1:10)
```

# Counting

```
## # A tibble: 10 x 3
##    word1     word2          n
##    <chr>     <chr>      <int>
##  1 fake      news         169
##  2 border    security     110
##  3 witch     hunt         108
##  4 president trump         80
##  5 white     house         67
##  6 southern  border        65
##  7 total     endorsement    56
##  8 news      media         54
##  9 north     korea         51
## 10 crooked   hillary       47
```

# Weighing

Tf-Idf

$$tfidf\,(\text{term}) = n_{\text{term}} \cdot \ln\left(\frac{n_{\text{documents}}}{n_{\text{documents containing term}}}\right)$$

# Weighing

```
tweet_words = trump_tweets %>%
  mutate(tweet = row_number()) %>%
  # split a column into tokens
  # using the tokenizers package
  unnest_tokens(word, text) %>%
  # remove stop_words
  anti_join(stop_words) %>%
  # count up the words within each tweet
  count(tweet, word, sort=TRUE)

total_words = tweet_words %>%
  group_by(tweet) %>%
  summarize(total = sum(n))

left_join(tweet_words, total_words) %>%
  bind_tf_idf(word, tweet, n) %>%
  arrange(desc(tf_idf)) %>%
  select(word, tf_idf) %>% distinct() %>% head()
```

# Weighing

```
## Joining, by = "word"
```

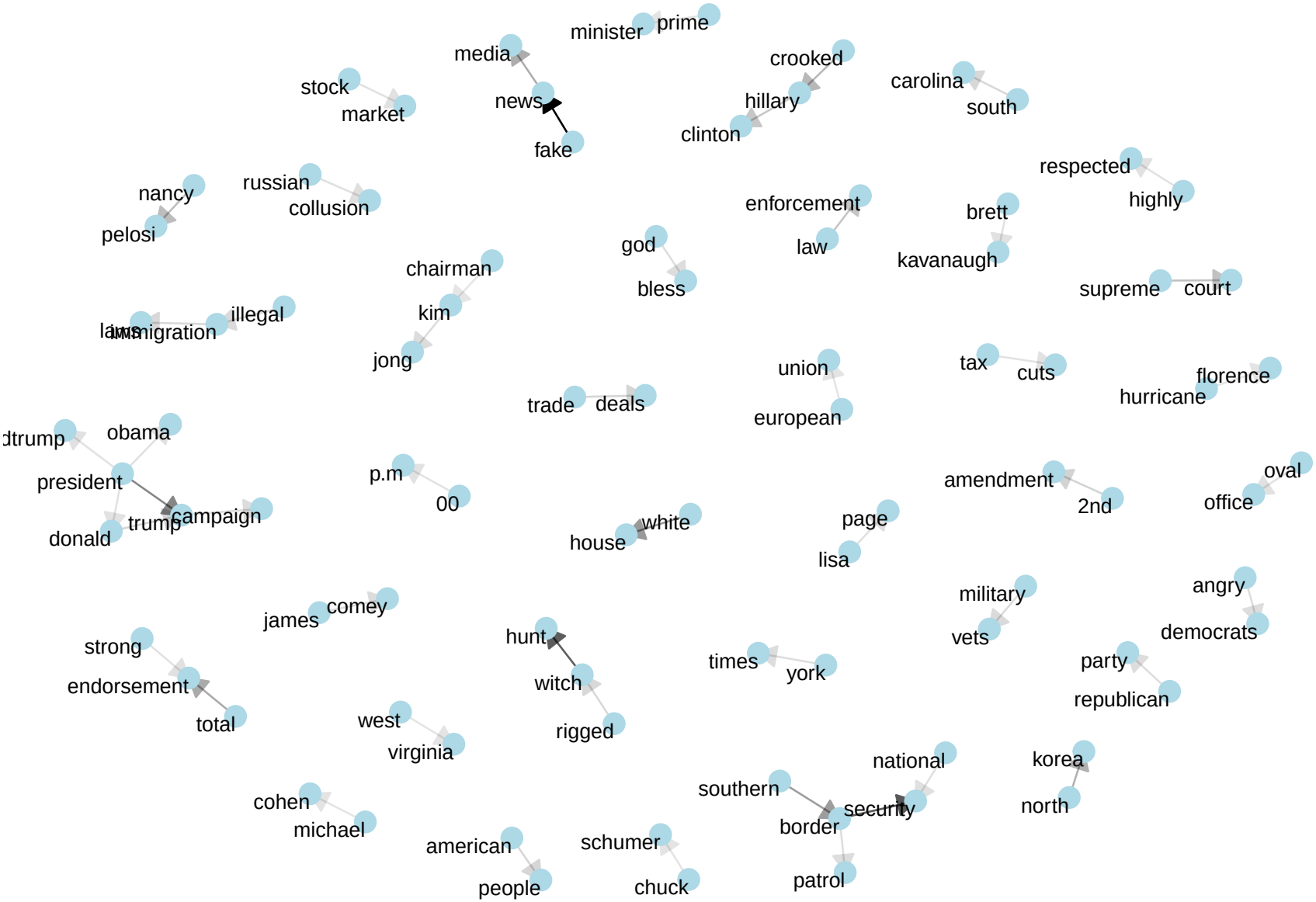
```
## Joining, by = "tweet"
```

```
## # A tibble: 6 x 2
##   word                  tf_idf
##   <chr>                  <dbl>
## 1 lrihendry               8.03
## 2 jt                      8.03
## 3 holocaustmemorialday    8.03
## 4 spot                    8.03
## 5 remembering41           8.03
## 6 prevail                 8.03
```

# Plotting

```
trump_tweets %>%
    unnest_tokens(bigram, text,
                  token = "ngrams", n = 2) %>%
    separate(bigram,
             c("word1", "word2"), sep = " ") %>%
    filter(!word1 %in% stop_words$word,
           !word2 %in% stop_words$word) %>%
    count(word1, word2, sort=TRUE) %>%
    filter(n > 15) %>%
    visualize_bigrams()
```

# Plotting


minister
prime
media
stock
market
news
fake
crooked
hillary
clinton
carolina
south
respected
highly
nancy
pelosi
russian
collusion
enforcement
law
brett
kavanaugh
god
bless
chairman
kim
jong
supreme
court
illegal
immigration
union
european
tax
cuts
florence
hurricane
trade
deals
dtrump
obama
president
donald
trump
campaign
p.m
00
amendment
2nd
oval
office
house
white
page
lisa
military
vets
angry
democrats
james
comey
strong
endorsement
total
hunt
witch
rigged
times
york
party
republican
west
virginia
national
security
korea
north
southern
border
patrol
cohen
michael
american
people
schumer
chuck

# Not Wordclouds

# Topic modelling

- LSA
- LDA
- STM
- `library(text2vec)` not `library(tm)`

# LDA

```
library(text2vec)
tokens = trump_tweets %>%
  tolower %>%
  word_tokenizer

it = itoken(tokens, progressbar = FALSE)

v = create_vocabulary(it,
                      stopwords = pull(stop_words, word)) %
  prune_vocabulary(term_count_min=2)

vectorizer = vocab_vectorizer(v)

dtm = create_dtm(it,
                 vectorizer,
                 type = "dgTMatrix")
```

# LDA

```
lda_model = LDA$new(n_topics = 5,
                    doc_topic_prior = 0.1,
                    topic_word_prior = 0.01)
doc_topic_distr =
  lda_model$fit_transform(x = dtm, n_iter = 1000,
                          convergence_tol = 0.001,
                          n_check_convergence = 25,
                          progressbar = FALSE)
```

# LDA

```
##        [,1]              [,2]       [,3]
##  [1,] "u.s"             "wall"     "trump"
##  [2,] "democrats"       "time"     "border"
##  [3,] "media"           "news"     "house"
##  [4,] "united"          "strong"   "democrats"
##  [5,] "don't"           "security" "hunt"
##  [6,] "vote"            "military" "borders"
##  [7,] "witch"           "total"    "deal"
##  [8,] "realdonaldtrump" "election" "jobs"
##  [9,] "world"           "vote"     "china"
## [10,] "hillary"         "hard"     "mueller"
```

# Word Embeddings


Figure 1: h/t The Morning Paper, A Colyer

# Word Embeddings

Mikolov et. al, 2013

`library(text2vec)`, or you can do it by hand with `library(tidytext)` (Julia Silge)

# Text classification

- Bag of words, Tf-Idf, or embeddings
- `library(caret)` SVM with linear kernel, or Logistic Regression
- Multi-label classification `library(mlr)`

# Neural Network thoughts

- Word embeddings + conv net
- RNN, LSTM, attention, BERT, etc.
- rKeras...