Web Scraping and Word Frequency

Julian Winkel

Ladislaus von Bortkiewicz Chair of Statistics

C.A.S.E.-Center for Applied Statistics and Economics

International Research Training Group

Humboldt-Universität zu Berlin

lvb.wiwi.hu-berlin.de

www.case.hu-berlin.de

irtg1792.hu-berlin.de

Donald Trump's Inaugural Speech

Source: CNN

Donald Trump's entire inaugural address 17:17

(CNN) — *As prepared for delivery*

Chief Justice Roberts, President Carter, President Clinton, President Bush, President Obama, fellow Americans, and people of the world: Thank you.

```
class="metadata"><p class="metadata__source"><span class="metadata__source-name"></span></p></div></div><div class="media__caption el__storyelement__title"><span
class="el__storyelement__header">Donald Trump's entire inaugural address</span> <span class="el__storyelement__gray">17:17</span></div></div></div></div></section></div>
<div class="pg-rail-tall__body" itemprop="articleBody"><section class="zn zn-body-text zn-body zn--idx-0 zn--ordinary zn-has-multiple-containers zn-has-79-containers"
data-eq-pts="xsmall: 0, medium: 460, large: 780, full16x9: 1100" id="body-text" data-vr-zone="zone-1-0" data-zone-label="bodyText" data-containers="79" id="body-text"
data-zn-id="body-text"><div class="1-container"><div class="el__leafmedia el__leafmedia--sourced-paragraph"><p class="zn-body__paragraph speakable"><cite class="el-
editorial-source"> (CNN)</cite><em>As prepared for delivery</em></p></div><div class="zn-body__paragraph speakable">Chief Justice Roberts, President Carter, President
Clinton, President Bush, President Obama, fellow Americans, and people of the world: Thank you.</div><div class="ad ad--epic ad--mobile" data-ad-text="show"><div
id="ad_rect_atf_02" class="ad-ad_rect_atf_02 ad-refresh-default"></div></div><ul class="cn cn-list-hierarchical-xs cn--idx-4 cn-zoneAdContainer"></ul><div class="zn-
body__paragraph speakable">We, the citizens of America, are now joined in a great national effort to rebuild our country and to restore its promise for all of our people.
</div><div class="zn-body__paragraph">Together, we will determine the course of America and the world for years to come.</div><div class="zn-body__paragraph">We will face
challenges. We will confront hardships. But we will get the job done.</div><div class="zn-body__read-more"><div class="read-more-gradient"></div><div class="read-more-
button"><div class="read-more-link" id="js-body-read-more">Read More</div></div></div><div class="zn-body__read-all"><div class="zn-body__paragraph">Every four years, we
gather on these steps to carry out the orderly and peaceful transfer of power, and we are grateful to President Obama and First Lady Michelle Obama for their gracious aid
throughout this transition. They have been magnificent.</div><div class="zn-body__paragraph">Today's ceremony, however, has very special meaning. Because today we are not
merely transferring power from one administration to another, or from one party to another -- but we are transferring power from Washington, D.C. and giving it back to
you, the American People.</div><div class="zn-body__paragraph"><a href="http://www.cnn.com/interactive/2017/01/politics/words-from-the-past/">Words from the past:
```

http://www.cnn.com/2017/01/20/politics/trump-inaugural-address/

What to the Slave is the Fourth of July?

Frederick Douglass
July 5, 1852

Mr. President, Friends and Fellow Citizens:

He who could address this audience without a quailing sensation, has stronger nerves than I have. I do not remember ever to have appeared as a speaker before any assembly more shrinkingly, nor with greater distrust of my ability, than I do this day. A feeling has crept over me, quite unfavorable to the exercise of my limited powers of speech. The task before me is one which requires much previous thought and study for its proper performance. I know that apologies of this sort are generally considered flat and unmeaning. I trust, however, that mine will not be so considered. Should I seem at ease, my appearance would much misrepresent me. The little experience I have had in addressing public meetings, in country schoolhouses, avails me nothing on the present occasion.

http://teachingamericanhistory.org/library/document/what-to-the-slave-is-the-fourth-of-july/

Text Extraction

```
import pandas as pd
import bs4
import urllib

# Define URL
url = "http://www.cnn.com/2017/01/20/politics/trump-inaugural-address/"

raw_html    = urllib.request.urlopen(url)
parsed_html = bs4.BeautifulSoup(raw_html, "lxml")
text        = parsed_html.find_all("div", class_ = "zn-body__paragraph")

# Speech stored in list
textl = []
for i in text:
    textl.append(i.get_text())

# Convert to string
cleantextprep = str(textl)
```

Regular Expressions

- Character Sequence
- Define a pattern
- Find and Replace

```
import re

# Clean Speech Text
expression     = "[^a-zA-Z0-9 ]"       # keep only letters, numbers and whitespace
cleantextCAP = re.sub(expression, "", cleantextprep)  # replace empty string
cleantext       = cleantextCAP.lower()

# Create Dictionary
dat = list(cleantext.split())
dict1 = {}
for i in range(len(dat)):
    print(i)
    word = dat[i]
    dict1[word] = dat.count(word)
```

Stopwords

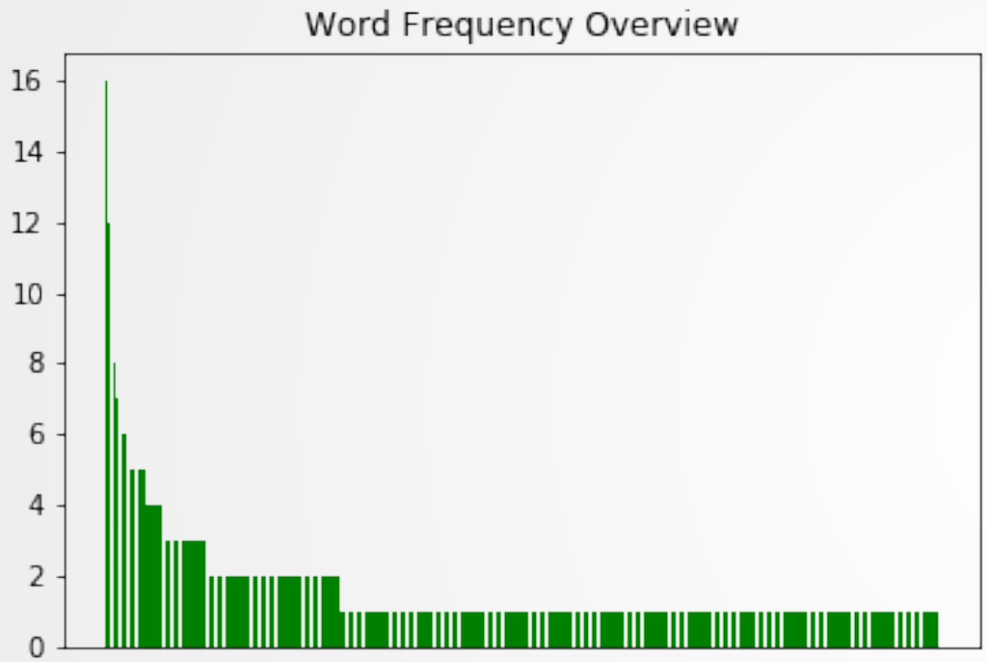
Applying a filter on natural language

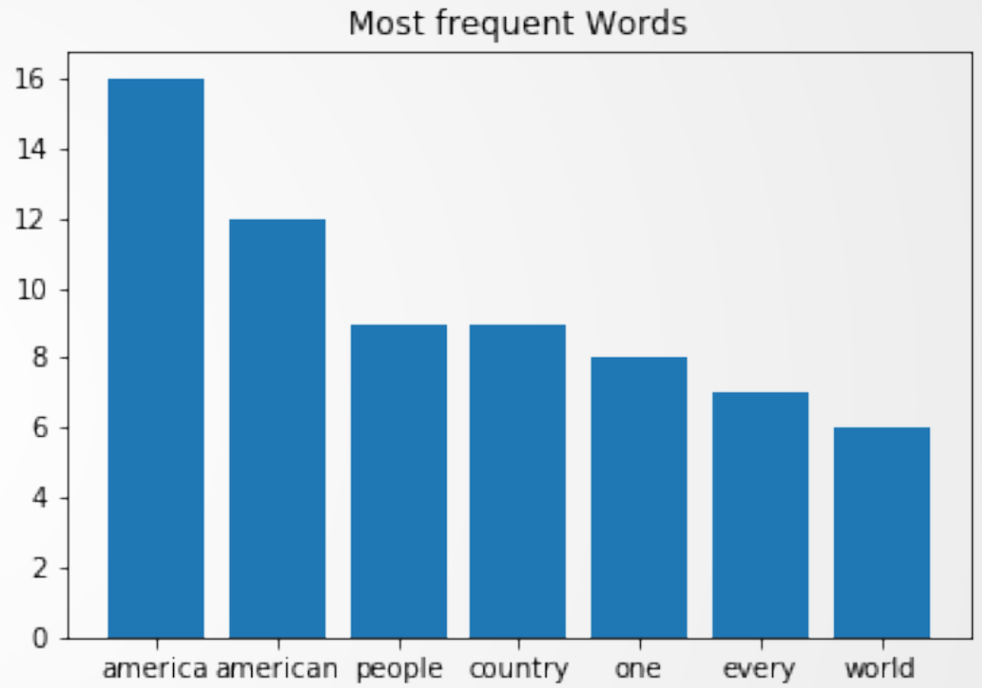
```
from  nltk.corpus import stopwords

# Unsorted speech constituents in dictionary as dict1
keys             = list(dict1)
filtered_words = [word for word in keys if word not in stopwords.words('english')]
dict2            = dict((k, dict1[k]) for k in filtered_words if k in filtered_words)
```

Sequence Selection

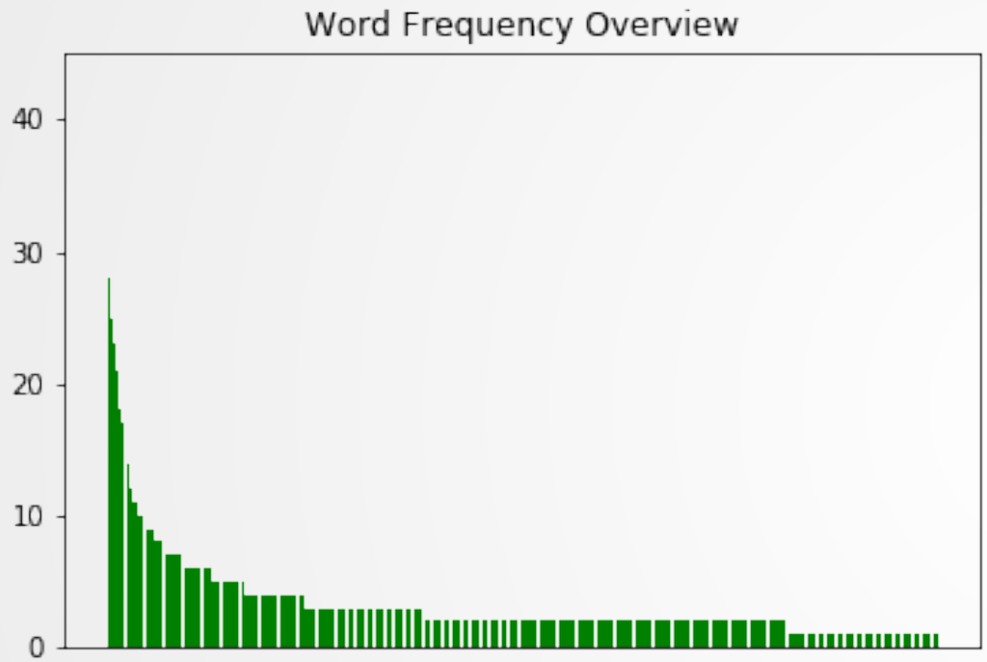
```
# Finding word sequences in a dictionary ordered by frequency
def SequenceSelection(dictionary, length, startindex = 0):

    # Check Input
    if length > len(dictionary):
        return print("input length is too long");
    else:
        d = dictionary
        items = [(v, k) for k, v in d.items()]
        items.sort()
        items.reverse()
        itemsOut = [(k, v) for v, k in items]

        highest = itemsOut[startindex:startindex + length]
        dd = dict(highest)
        wanted_keys = dd.keys()
        dictshow = dict((k, d[k]) for k in wanted_keys if k in d)
       return dictshow;

dictshow = SequenceSelection(dictionary = dict2, length = 7, startindex = 0)
```

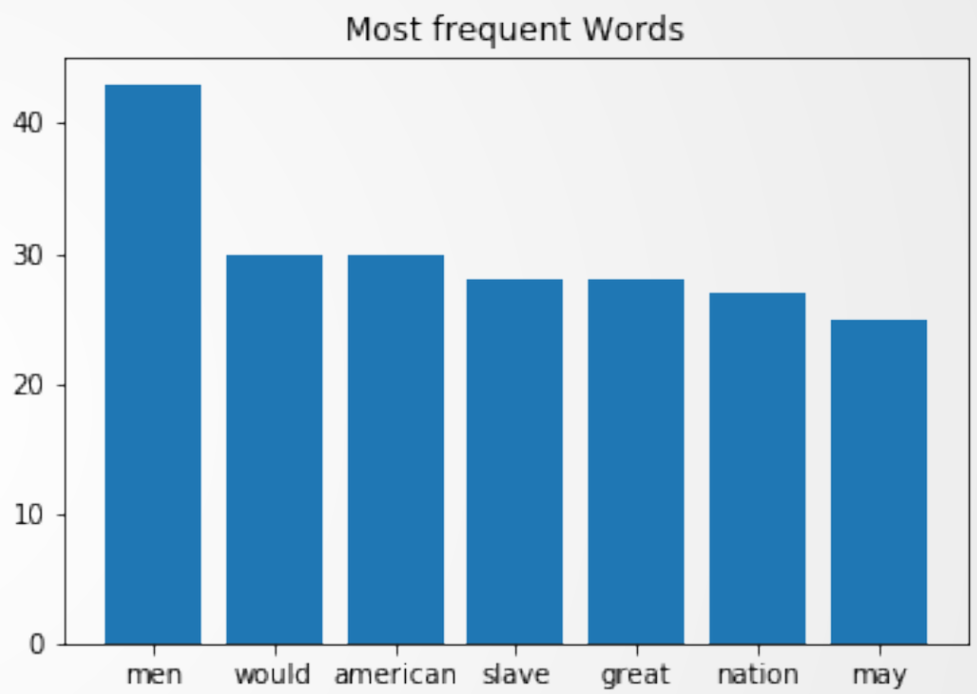
Word Frequency Plots - Trump's Inaugural Speech





Word Frequency Plots - Frederick Douglass





Related frequent words among the speeches:

America, Nation, People, Country

Word Cloud I

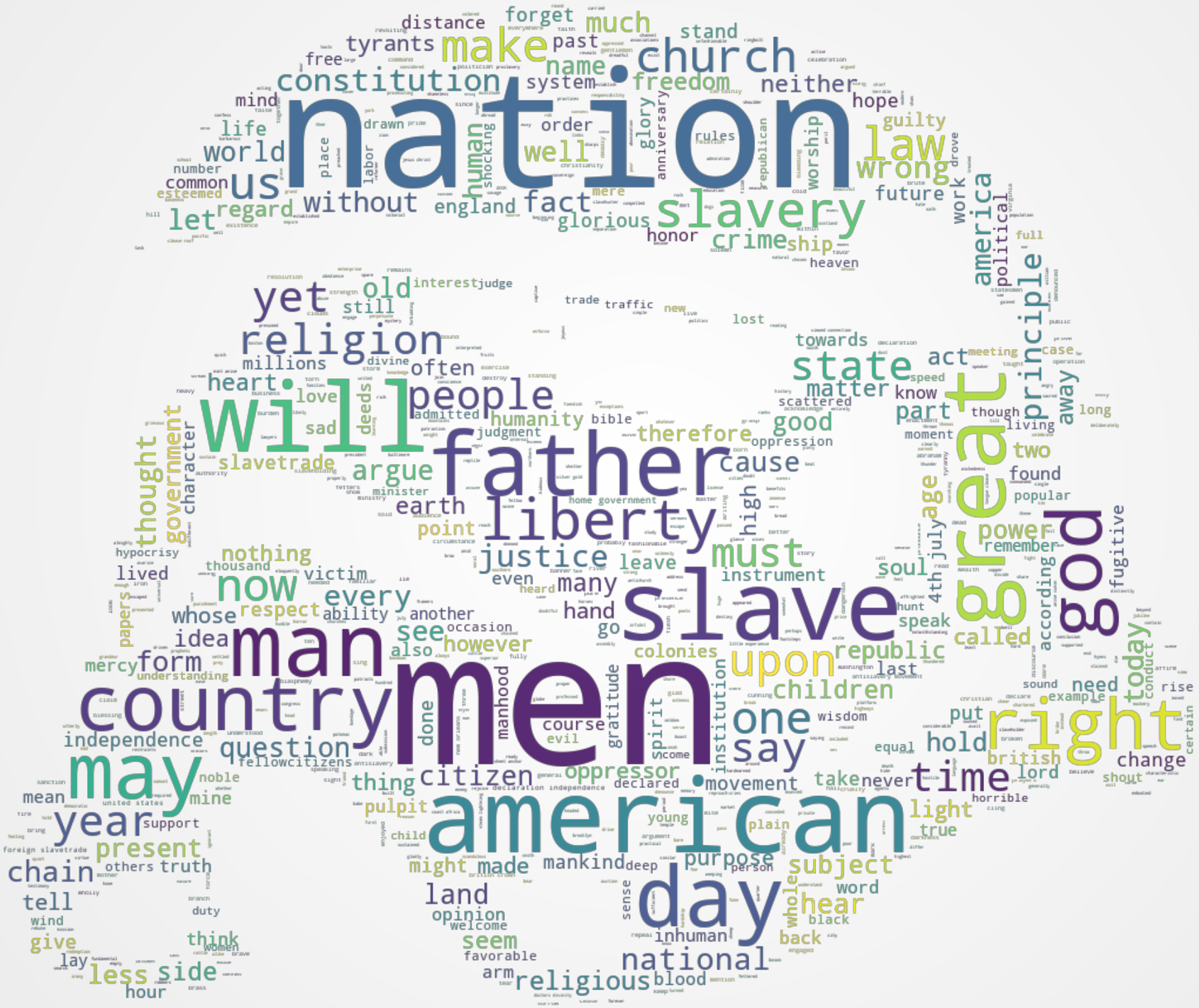
```
import os
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

d = os.environ['USERPROFILE'] + '\pyning'

# Read the Speech
text = open(path.join(d, 'TrumpSpeech.txt')).read()

# Mask
stormtrooper_mask = np.array(
                        Image.open(
                        path.join(d, "stormtrooper_mask.png")
))

# Optional additional stopwords
stopwords = set(STOPWORDS)
stopwords.add("said")
```

Word Cloud II

```
# Construct Word Cloud

wc = WordCloud( max_words = 1000,
                mask = stormtrooper_mask,
                stopwords = stopwords,
                mode = 'RGBA',
                background_color = None) # transparency

# Apply on Speech as text
wc.generate(text)

# Store to File, d as directory
wc.to_file(path.join(d, "a_new_hope.png"))
```

Custom Word Cloud - Trump's Inaugural Speech

Custom Word Cloud - Frederick Douglass



Sentiment Analysis I - Overview

- Extract and study information from texts
- Match extracted words to dictionary
- Define measures for text intention

... McDonald's has an obesity **problem** that continues to get **worse**. And that's nothing to do with the food itself, but rather the huge menus that can now double as medieval fortification. For perspective, the chain's menu has grown 70% since 2007. And while more offerings might seem **like** a **good** thing, large menus result in **slower** service and more flare-ups between franchisees and the corporation.
Bloated menus raise inventory costs for smaller franchisees and **lead** to lower profit margins. The McDonald's corporate franchise fee is based upon sales instead of profits, making it a smaller **concern** for the company overall. ...

3 **positive words** and 5 **negative words**

Sentiment Analysis II - Harvard IV

- English and French
- 83 unique categories and various combinations
- Categories: Male-role, affection, distress, pleasure, medical
- 'Devastate' - Attributes: Negative, hostile, strong
- 'Glimmer' - Attributes: Positive, passive, strong

Sentiment Analysis III - Application

How to apply the dictionary on a text?

1) Split text into constituent words
2) Match words to dictionary
3) Count attributes

N = {Positive, Negative, Neutral}
Polarity = (Positive - Negative)/(Positive + Negative)
Subjectivity = (Positive + Negative)/N

Sentiment Analysis IIII

```
import pysentiment as ps

#  Trump's Speech as cleantext
hiv4     = ps.HIV4()                    # Use Harvard IV
tokens = hiv4.tokenize(cleantext)    # split string into constituents
score   = hiv4.get_score(tokens)
print(score)
```

Trump's Inaugural Speech

Positive: 133
Negative: 50
Polarity: 0.45
Subjectivity: 0.405

What to the Slave is the Fourth of July?

Positive: 783
Negative: 593
Polarity: 0.13
Subjectivity: 0.38

Latent Dirichlet Allocation (LDA)

- Topic Model: Match words to topics
- Document may exhibit several latent topics
- Each topic is a distribution over a vocabulary
- Which topic category is most likely to generate these words?
- Assume generative process

LDA - Trump's Speeches

- Analyze 12 Speeches from 2017
- ~ 1 MB all on GitHub
- 170,000 Words

```
import os
import re

d = os.environ['USERPROFILE'] + '\pyning'

# Read all Speeches
text = open(path.join(d, 'speeches.txt'), encoding = 'utf8').read()
doc_complete = str.split(text, sep = 'Speech')

doc_out = []
for l in doc_complete:
    cleantextprep = str(l)
    expression = '[^a-zA-Z ]'
    cleantextCAP = re.sub(expression, ' ', cleantextprep)
    bound = ' '.join(cleantext)
    doc_out.append(bound)
```

LDA - Trump's Speeches II

- All text documents combined is known as the corpus.
- Convert corpus into a matrix representation
- Document Term Matrix Dimension: (Term x Documents).

	Word 1	Word 2	Word ...	Word N
Article 1	0	2	...	1
Article 2	2	2	...	0
Article 3	3	5	...	1
...	...	...	...	...
Article M	0	0	...	1

https://appliedmachinelearning.wordpress.com/2017/09/28/topic-modelling-part-2-discovering-topics-from-articles-with-latent-dirichlet-allocation/

LDA - Trump's Speeches III

- Apply LDA Algorithm - Reduce Dimension

	Topic 1	Topic 2	Topic ...	Topic K
Article 1	$P(t_1\|a_1)$	$P(t_2\|a_1)$	...	$P(t_K\|a_1)$
Article 2	$P(t_1\|a_2)$	$P(t_2\|a_2)$	...	$P(t_K\|a_2)$
Article 3	$P(t_1\|a_3)$	$P(t_2\|a_3)$	...	$P(t_K\|a_3)$
...	...	...	...	
Article M	$P(t_1\|a_M)$	$P(t_2\|a_M)$	...	$P(t_K\|a_M)$

- (Document x Topic)

	Word 1	Word 2	Word ...	Word N
Topic 1	$P(w_1\|t_1)$	$P(w_2\|t_1)$	...	$P(w_N\|t_1)$
Topic 2	$P(w_1\|t_2)$	$P(w_2\|t_2)$	...	$P(w_N\|t_1)$
Topic 3	$P(w_1\|t_3)$	$P(w_2\|t_3)$	...	$P(w_N\|t_1)$
...	...	...	...	...
Topic K	$P(w_1\|t_K)$	$P(w_2\|t_M)$	...	$P(w_N\|t_1)$

- (Topic x Term)

LDA - Trump's Speeches IV

- Apply Stopwords, remove Punctuation, Lemmatization

```
import gensim
from gensim import corpora

dictionary = corpora.Dictionary(doc_clean) # assign words to index

# Create Document Term Matrix
doc_term_matrix = [dictionary.doc2bow(doc) for doc in doc_clean]

Lda = gensim.models.ldamodel.LdaModel

ldamodel = Lda(doc_term_matrix, num_topics = 2, id2word = dictionary,
               passes = 100)

print(ldamodel.print_topics(num_topics = 2, num_words = 3)
```

LDA - Trump's Speeches V

- Output - 2 Topics x 3 Words
- Topic 1: 0.016 * 'people' + 0.012 * 'dont' + 0.010 * 'want'
- Topic 2: 0.006 * 'must' + 0.005 * 'foreign' + 0.004 * 'policy'
- Topic 1: Voter's Desire
- Topic 2: Foreign Affairs (China, Mexico)

Word Cloud - Trump's Speeches

Sentiment Analysis Trump's Speeches

N = {Positive, Negative, Neutral}
Polarity = (Positive - Negative)/(Positive + Negative)
Subjectivity = (Positive + Negative)/N

Positive: 10539
Negative: 6926
Polarity: 0.2068
Subjectivity: 0.3316

Web Scraping and Word Frequency

Julian Winkel

Ladislaus von Bortkiewicz Chair of Statistics

C.A.S.E.-Center for Applied Statistics and Economics

International Research Training Group

Humboldt-Universität zu Berlin

lvb.wiwi.hu-berlin.de

www.case.hu-berlin.de

irtg1792.hu-berlin.de