Business Analytics Programming

Lab 4 - Twitter Analytics

Dr. Wajahat Gilani

Rutgers Business School

March 28, 2019

Textblob Download

```
1 import pip
2
3 !pip install textblob
4 !python -m textblob.download_corpora
5
6 from textblob import TextBlob
```

Pip is used to download 3rd-party python libraries. Lines 3 and 4 will download the textblob library as well as a list of words and their identifications.

Textblob Primary Functions

```python
tx = df.loc[0,'full_text']
blob = TextBlob(tx)
blob.tags
blob.sentences[0].words
blob.noun_phrases
blob.ngrams(3)
blob.correct()
blob.words[3].spellcheck()
blob.detect_language()
blob.translate(to= 'ar')
```

tx:

'The Fake News Media has lost tremendous credibility with its corrupt coverage of the illegal Democrat Witch Hunt of your all time favorite duly elected President, me! T.V. ratings of CNN &amp; MSNBC tanked last night after seeing the Mueller Report statement. @FoxNews up BIG!'

Textblob Primary Functions - Tags

```
blob.tags
```

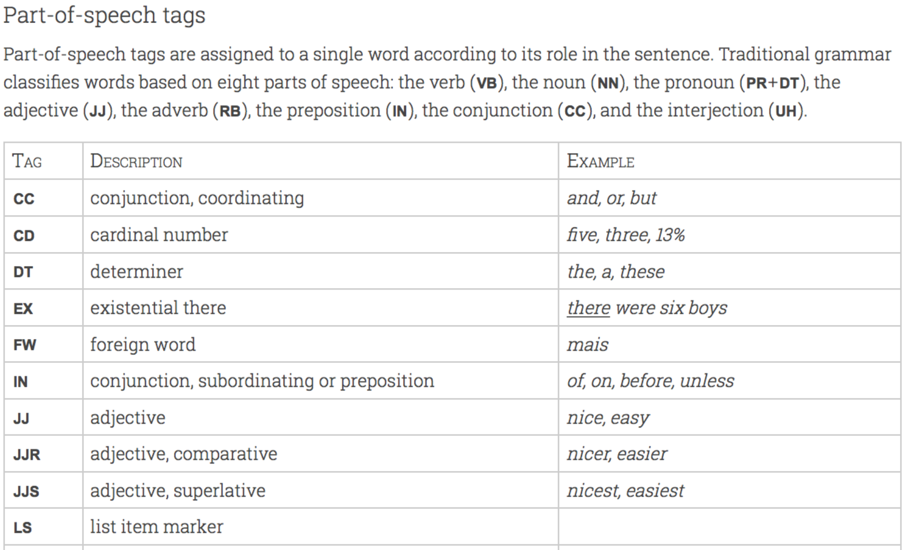
```
[('The', 'DT'), ('Fake', 'NNP'), ('News', 'NNP'), ('Media', 'NNP'), ('has',
'VBZ'), ('lost', 'VBN'), ('tremendous', 'JJ'), ('credibility', 'NN'), ('with',
'IN'), ('its', 'PRP$'), ('corrupt', 'JJ'), ('coverage', 'NN'), ('of', 'IN'),
('the', 'DT'), ('illegal', 'JJ'), ('Democrat', 'NNP'), ('Witch', 'NNP'),
('Hunt', 'NNP'), ('of', 'IN'), ('your', 'PRP$'), ('all', 'DT'), ('time', 'NN'),
('favorite', 'JJ'), ('duly', 'RB'), ('elected', 'VBN'), ('President', 'NNP'),
('me', 'PRP'), ('T.V', 'NNP'), ('ratings', 'NNS'), ('of', 'IN'), ('CNN',
'NNP'), ('&', 'CC'), ('amp', 'NN'), ('MSNBC', 'NNP'), ('tanked', 'VBD'),
('last', 'JJ'), ('night', 'NN'), ('after', 'IN'), ('seeing', 'VBG'), ('the', 'DT'),
('Mueller', 'NNP'), ('Report', 'NNP'), ('statement', 'NN'), ('@', 'NN'),
('FoxNews', 'NNP'), ('up', 'RB'), ('BIG', 'NNP')]
```

```
blob.tags[0]
```

```
('The', 'DT')
```

Textblob Primary Functions

https://www.clips.uantwerpen.be/pages/mbsp-tags



Part-of-speech tags

Part-of-speech tags are assigned to a single word according to its role in the sentence. Traditional grammar classifies words based on eight parts of speech: the verb (**VB**), the noun (**NN**), the pronoun (**PR+DT**), the adjective (**JJ**), the adverb (**RB**), the preposition (**IN**), the conjunction (**CC**), and the interjection (**UH**).

Tag	Description	Example
CC	conjunction, coordinating	*and, or, but*
CD	cardinal number	*five, three, 13%*
DT	determiner	*the, a, these*
EX	existential there	*there were six boys*
FW	foreign word	*mais*
IN	conjunction, subordinating or preposition	*of, on, before, unless*
JJ	adjective	*nice, easy*
JJR	adjective, comparative	*nicer, easier*
JJS	adjective, superlative	*nicest, easiest*
LS	list item marker	

Textblob Primary Functions - Sentences/Words

```
blob.sentences
```

```
[Sentence(”The Fake News Media has lost tremendous credibility with its
corrupt coverage of the illegal Democrat Witch Hunt of your all time
favorite duly elected President, me!”), Sentence(”T.V.”),
Sentence(”ratings of CNN &amp; MSNBC tanked last night after seeing
the Mueller Report statement.”), Sentence(”@FoxNews up BIG!”)]
```

```
blob.sentences[0]
```

```
Sentence(”The Fake News Media has lost tremendous credibility with its
corrupt coverage of the illegal Democrat Witch Hunt of your all time
favorite duly elected President, me!”)
```

```
blob.sentences[0].words
```

```
WordList(['The', 'Fake', 'News', 'Media', 'has', 'lost', 'tremendous',
'credibility', 'with', 'its', 'corrupt', 'coverage', 'of', 'the', 'illegal',
'Democrat', 'Witch', 'Hunt', 'of', 'your', 'all', 'time', 'favorite', 'duly',
'elected', 'President', 'me'])
```

Textblob Primary Functions - Sentences/Words

```
blob.sentences[0]
```

```
Sentence("The Fake News Media has lost tremendous credibility with its
corrupt coverage of the illegal Democrat Witch Hunt of your all time
favorite duly elected President, me!")
```

```
blob.sentences[0].words
```

```
WordList(['The', 'Fake', 'News', 'Media', 'has', 'lost', 'tremendous',
'credibility', 'with', 'its', 'corrupt', 'coverage', 'of', 'the', 'illegal',
'Democrat', 'Witch', 'Hunt', 'of', 'your', 'all', 'time', 'favorite', 'duly',
'elected', 'President', 'me'])
```

```
blob.sentences[0].words[0]
```

```
'The'
```

Textblob Primary Functions - Sentences/Words

```
blob.sentences[0]
```

```
Sentence(”The Fake News Media has lost tremendous credibility with its
corrupt coverage of the illegal Democrat Witch Hunt of your all time
favorite duly elected President, me!”)
```

```
blob.sentences[0].words
```

```
WordList(['The’, 'Fake’, 'News’, 'Media’, 'has’, 'lost’, 'tremendous’,
'credibility’, 'with’, 'its’, 'corrupt’, 'coverage’, 'of’, 'the’, 'illegal’,
'Democrat’, 'Witch’, 'Hunt’, 'of’, 'your’, 'all’, 'time’, 'favorite’, 'duly’,
'elected’, 'President’, 'me’])
```

```
blob.sentences[0].words[0]
```

```
'The’
```

Textblob Primary Functions - Ngrams

```python
blob.ngrams(3)
```

```
[WordList(['The', 'Fake', 'News']), WordList(['Fake', 'News', 'Media']),
WordList(['News', 'Media', 'has']), WordList(['Media', 'has', 'lost']),
WordList(['has', 'lost', 'tremendous']), WordList(['lost', 'tremendous',
'credibility']), WordList(['tremendous', 'credibility', 'with']),
WordList(['credibility', 'with', 'its']), WordList(['with', 'its', 'corrupt']),
WordList(['its', 'corrupt', 'coverage']), WordList(['corrupt', 'coverage',
'of']), WordList(['coverage', 'of', 'the']), WordList(['of', 'the', 'illegal']),
WordList(['the', 'illegal', 'Democrat']), WordList(['illegal', 'Democrat',
'Witch']), WordList(['Democrat', 'Witch', 'Hunt']), WordList(['Witch',
'Hunt', 'of']), WordList(['Hunt', 'of', 'your']), WordList(['of', 'your', 'all']),
WordList(['your', 'all', 'time']), WordList(['all', 'time', 'favorite']),
WordList(['time', 'favorite', 'duly']), WordList(['favorite', 'duly', 'elected']),
WordList(['duly', 'elected', 'President']), WordList(['elected', 'President',
'me']), WordList(['President', 'me', 'T.V']), WordList(['me', 'T.V',
'ratings']), WordList(['T.V', 'ratings', 'of']), WordList(['ratings', 'of',
'CNN']), WordList(['of', 'CNN', 'amp']), WordList(['CNN', 'amp',
```

Textblob Primary Functions - Ngrams

```
blob.ngrams(5)
```

```
[WordList(['The', 'Fake', 'News', 'Media', 'has']), WordList(['Fake',
'News', 'Media', 'has', 'lost']), WordList(['News', 'Media', 'has', 'lost',
'tremendous']), WordList(['Media', 'has', 'lost', 'tremendous',
'credibility']), WordList(['has', 'lost', 'tremendous', 'credibility', 'with']),
WordList(['lost', 'tremendous', 'credibility', 'with', 'its']),
WordList(['tremendous', 'credibility', 'with', 'its', 'corrupt']),
WordList(['credibility', 'with', 'its', 'corrupt', 'coverage']),
WordList(['with', 'its', 'corrupt', 'coverage', 'of']), WordList(['its',
'corrupt', 'coverage', 'of', 'the']), WordList(['corrupt', 'coverage', 'of',
'the', 'illegal']), WordList(['coverage', 'of', 'the', 'illegal', 'Democrat']),
WordList(['of', 'the', 'illegal', 'Democrat', 'Witch']), WordList(['the',
'illegal', 'Democrat', 'Witch', 'Hunt']), WordList(['illegal', 'Democrat',
'Witch', 'Hunt', 'of']), WordList(['Democrat', 'Witch', 'Hunt', 'of',
'your']), WordList(['Witch', 'Hunt', 'of', 'your', 'all']), WordList(['Hunt',
'of', 'your', 'all', 'time']), WordList(['of', 'your', 'all', 'time', 'favorite']),
WordList(['your', 'all', 'time', 'favorite', 'duly']), WordList(['all', 'time',
```

Textblob Primary Functions - Ngrams

```python
blob.ngrams(5)[0]
```

```
WordList(['The', 'Fake', 'News', 'Media', 'has'])
```

Textblob Primary Functions - Correct()

’The Fake News Media has lost tremendous credibility with its corrupt coverage of the illegal Democrat Witch Hunt of your all time favorite duly elected President, me! T.V. ratings of CNN &amp; MSNBC tanked last night after seeing the Mueller Report statement. @FoxNews up BIG!’

```
blob.correct()
```

TextBlob(”The Take News Media has lost tremendous credibility with its corrupt coverage of the illegal Democrat Pitch Aunt of your all time favorite duly elected President, me! T.V. rating of CNN &amp; MSNBC talked last night after seeing the Fuller Report statement. @FoxNews up BIG!”)

Textblob Primary Functions - Spellcheck()

```
blob.words[1].spellcheck()
```

```
[(’Take’, 0.4797507788161994), (’Make’, 0.3925233644859813), (’Sake’,
0.07554517133956386), (’Wake’, 0.02102803738317757), (’Lake’,
0.02102803738317757), (’Cake’, 0.004672897196261682), (’Rake’,
0.003115264797507788), (’Bake’, 0.001557632398753894), (’Jake’,
0.000778816199376947)]
```

```
blob.words[5].spellcheck()
```

```
[(’lost’, 1.0)]
```

Textblob Primary Functions - Language()

```
blob.detect_language()
```

```
'en'
```

```
blob.translate(to= 'fr')
```

```
TextBlob(”Fake News Media a perdu une énorme crédibilité avec sa
couverture corrompue de la chasse illégale à la sorcière démocrate de votre
grand favori, le président dûment élu, moi! Évaluations T.V. de CNN &
amp; MSNBC a fait une grimace la nuit dernière après avoir vu la
déclaration du rapport Mueller. @ FoxNews up BIG!”)
```

Textblob Primary Functions - Lemmatize

The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance:

am, are, is ⇒ be

car, cars, car's, cars' ⇒ car

```
verbs = []
for word, tag in blob.tags:
  if tag == 'VB':
    verbs.append(word.lemmatize())
```

verbs:[]

```
nouns = []
for word, tag in blob.tags:
  if tag == 'NN':
    nouns.append(word.lemmatize())
```

nouns:['credibility', 'coverage', 'time', 'amp', 'night', 'statement', '@']

Textblob Primary Functions - Lemmatize

```python
nounsp = []
for word, tag in blob.tags:
  if tag == 'NNP':
    nounsp.append(word.lemmatize())
```

nounsp:['Fake', 'News', 'Media', 'Democrat', 'Witch', 'Hunt', 'President', 'T.V', 'CNN', 'MSNBC', 'Mueller', 'Report', 'FoxNews', 'BIG']

Textblob Primary Functions - Sentiment

```
blob.sentiment
```

Answer:Sentiment(polarity=-0.0773809523809524, subjectivity=0.6666666666666666)

Subjectivity ranges from 1 to -1, 1 being positive, -1 being negative
Polarity ranges from 0 to 1, 0 being objective and 1 being subjective

```
blob.sentiment.polarity
blob.sentiment.subjectivity
```

```
-0.0773809523809524
0.6666666666666666
```

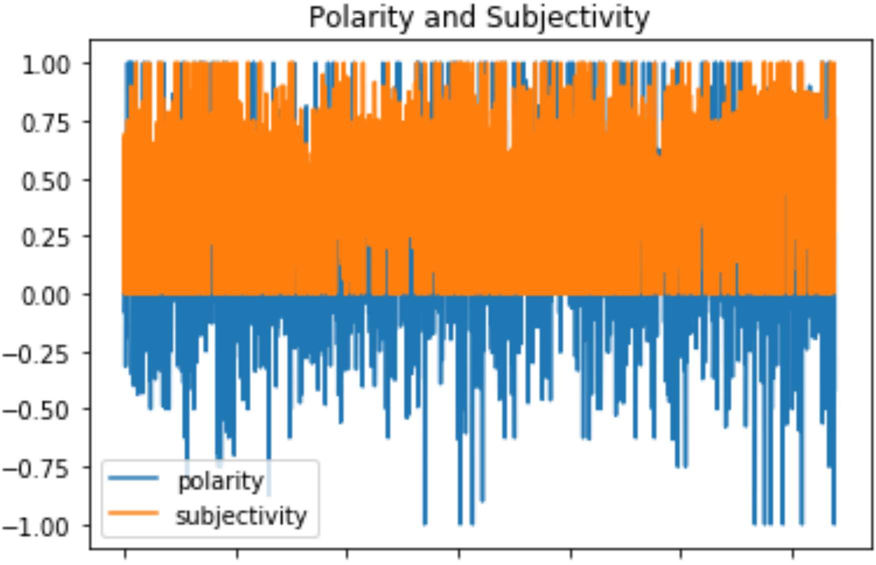
Textblob Primary Functions - Sentiment For All Tweets

```python
polarity=[]
subj=[]

for t in df.full_text:
  tx=TextBlob(t)
  polarity.append(tx.sentiment.polarity)
  subj.append(tx.sentiment.subjectivity)

polsubj = pd.DataFrame({'polarity': polarity,'subjectivity': subj})
```

```
    polarity  subjectivity
0  -0.077381      0.666667
1   0.000000      0.000000
2   0.000000      0.000000
3   0.160000      0.693333
4   0.000000      0.000000
5   0.200000      0.339286
6  -0.316667      0.300000
7   0.000000      0.000000
```

Textblob Primary Functions - Sentiment For All Tweets

```python
polsubj.plot(title='Polarity and Subjectivity')
```



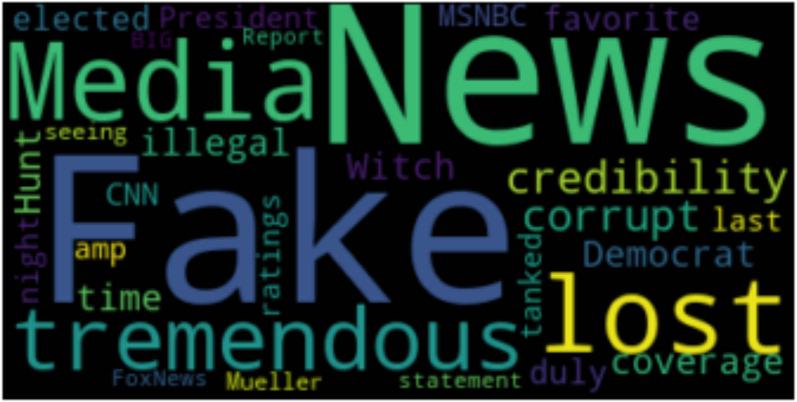
WordCloud

```python
import pip

#!pip install wordcloud
from nltk.corpus import stopwords

import matplotlib.pyplot as plt
from wordcloud import WordCloud

stop =stopwords.words('english')
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're",
"you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he',
'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its',
'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who',
'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was',
'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did',
'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while',
'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through',
'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in',
'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here',
```

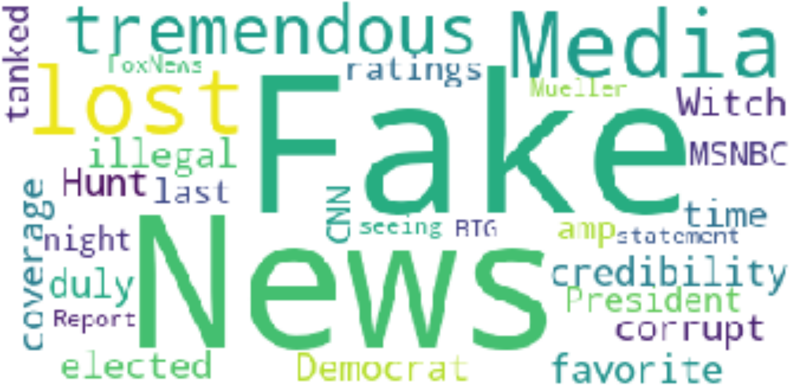
WordCloud

```python
wordcloud = WordCloud().generate(tx)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```



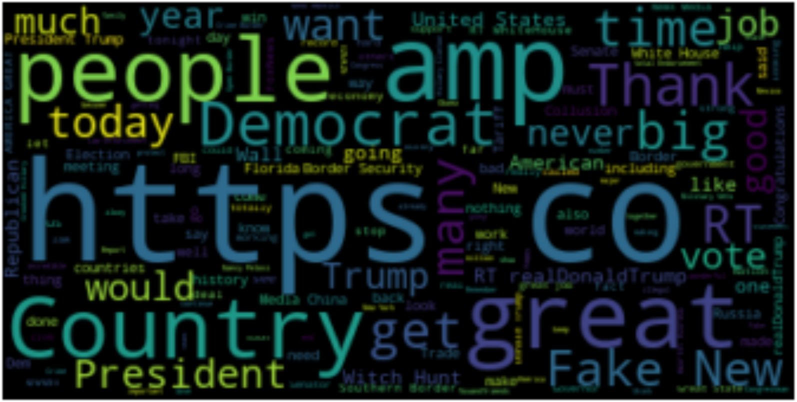
WordCloud

```
wordcloud2 = WordCloud(background_color="white",stopwords=
    stop).generate(tx)
plt.imshow(wordcloud2)
plt.axis("off")
plt.show()
```



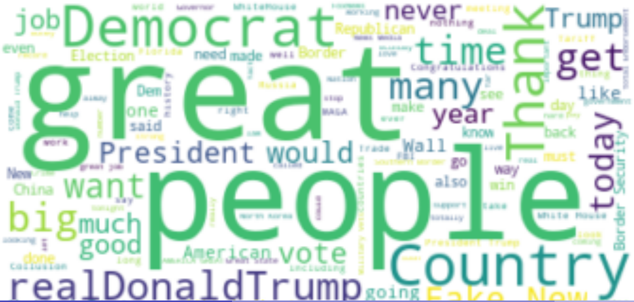
WordCloud

```
tx2=df.full_text.str.cat(sep=' ')

wordcloud3 = WordCloud(stopwords=stop).generate(tx2)
plt.imshow(wordcloud3, interpolation='bilinear')
plt.axis("off")
plt.show()
```



WordCloud

```
stop.append('RT')
stop.append('co')
stop.append('https')
stop.append('amp')

wordcloud4 = WordCloud(background_color="white",stopwords=
    stop, max_words=1000).generate(tx2)
plt.imshow(wordcloud4, interpolation='bilinear')
plt.axis("off")
plt.show()
```


great
people
Democrat
Thank
Country
realDonaldTrump
President
would
time
many
never
Trump
get
today
year
job
want
big
much
good
vote
American
like
Republican
Election
Border
need
made
Dem
one
said
Wall
China
New
Fake
News
going
back
must
day
know
make
see
also
way
go
Security
Congratulations
Russia
done
collusion
even