ASSIGNMENT 6 - Topic Modeling

You have been provided with a pickle file, containing 100 news articles about some company. Use appropriate topic modeling technique to identify top N most important topics.

Import the Necessary Packages

In [1]:

```
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
warnings.filterwarnings("ignore", category=DeprecationWarning)
import time
import math
import re
from textblob import TextBlob
import pandas as pd

import nltk as nltk
from nltk.corpus import stopwords
from nltk.stem.wordnet import WordNetLemmatizer

import string

import warnings
warnings.filterwarnings(action='ignore', category=UserWarning, module='gensim')

import gensim
from gensim import corpora, models
from gensim.models.ldamulticore import LdaMulticore
import pyLDAvis.gensim
```

Load the Data

In [2]:

```
directory = 'C://Users//mjdun//Desktop//NLP//Assignments//'
file= 'webhose_cat.pkl'
data=pd.read_pickle(directory+file)
data.describe(include='all')
```

Out[2]:

	crawled	language	text	title	url
count	100	100	100	100	100
unique	99	6	99	75	100
top	2018-01-30T19:39:22.011+02:00	english	Plants can thrive with no care at all in Wardi...		http://omgili.com/ri/.0rSU5LtMgxRirvkm0q5zYu0T...
freq	2	95	2	16	1

We see that there are six distinct languages. We will use only English language articles.

In [3]:

```
data = data[data['language']=='english']
```

Let us see what these articles look like.

In [4]:

```
data.iloc[0:5]
```

Out[4]:

	crawled	language	text	title	url
0	2018-01-30T18:28:45.012+02:00	english	Avery Dennison's (AVY) Q4 results are likely t...	IRobot downgraded to neutral from buy at Sidot...	http://omgili.com/ri/.wHSUbtEfZQRfU.5KUm1RkeXy...
2	2018-01-30T18:29:40.000+02:00	english	Tuggers and Topper Industrial Carts Help Trans...	Tuggers and Topper Industrial Carts Help Trans...	http://omgili.com/ri/jHIAmI4hxg.zDiulpymXqU_n4...
3	2018-01-30T18:30:05.007+02:00	english	Currently adding the following games:\n100 (by...		http://omgili.com/ri/.0rSU5LtMgyggHgoOVy9TMDWT...
4	2018-01-30T18:30:05.013+02:00	english	Quote: : » Currently adding the following game...		http://omgili.com/ri/.0rSU5LtMgyggHgoOVy9TMDWT...
5	2018-01-30T18:30:05.014+02:00	english	Quote: : » Currently adding the following game...		http://omgili.com/ri/.0rSU5LtMgyggHgoOVy9TMDWT...

Only some of these columns are relevant to our analysis, specifically the 'text' and 'title' columns. We care about both for purpose of Topic Modeling (there is a lot of information about the topic of a document in the title). Let us combine the text and title for each article and then put that value into a list.

In [5]:

```
articles=[]
for i in range(len(data)):
    combined = ' '.join([data['title'].iloc[i], data['text'].iloc[i]])
    articles.append(combined)
```

Present top N most important topics in these news articles

We will use **Latent Dirichlet Analysis** to generate our topics, but first some cleaning of each individual article (title and text combined) is required. Specifically, we will:
-remove stop words
-remove punctuation
-normalize the text through lemmatization

In [6]:

```
stop = set(stopwords.words('english'))
exclude = set(string.punctuation)
lemma = WordNetLemmatizer()
def clean(doc):
    stop_free = " ".join([i for i in doc.lower().split() if i not in stop])
    punc_free = ''.join(ch for ch in stop_free if ch not in exclude)
    normalized = " ".join(lemma.lemmatize(word) for word in punc_free.split())
    return normalized

articles_clean = [clean(doc).split() for doc in articles]
```

Then we create a dictionary from our corpus where every unique word is assigned an index. Then we convert our corpus into a Document Term Matrix using that dictionary.

In [7]:

```
# Creating the term dictionary of our corpus, where every unique term is assigned an index.
dictionary = corpora.Dictionary(articles_clean)

# Converting list of documents (corpus) into Document Term Matrix using dictionary prepared above.
doc_term_matrix = [dictionary.doc2bow(doc) for doc in articles_clean]

# Creating the object for LDA model using gensim library
Lda = gensim.models.ldamodel.LdaModel
```

Select N to identify relevant topics, but minimize duplication

In Topic Modeling one must select both the number of topics returned from the corpus (how many topics can we divide the corpus into) and the number of words in each topic (how many words we use to describe each topic). If we use too many topics to describe the corpus there ends up being overlap between topics. If we use too many words to describe the topic the description becomes non-sensical.

We will start with *three topics*.

In [8]:

```
three_model = Lda(doc_term_matrix, num_topics=3, id2word=dictionary, passes=50)
print(*three_model.print_topics(num_topics=3, num_words=3), sep='\n')
```

```
(0, '0.009*"tax" + 0.008*"u" + 0.006*"jan"')
(1, '0.009*"caterpillar" + 0.005*"share" + 0.004*"product"')
(2, '0.016*"market" + 0.009*"plant" + 0.006*"case"')
```

These topics are not especially understandable. If we add more words to the topics they become somewhat more understandable.

In [9]:

```
print(*three_model.print_topics(num_topics=3, num_words=6), sep='\n')
```

```
(0, '0.009*"tax" + 0.008*"u" + 0.006*"jan" + 0.004*"company" + 0.004*"inc" + 0.004*"year"')
(1, '0.009*"caterpillar" + 0.005*"share" + 0.004*"product" + 0.004*"company" + 0.004*"stock" + 0.004*"new"')
(2, '0.016*"market" + 0.009*"plant" + 0.006*"case" + 0.005*"median" + 0.005*"estimate" + 0.005*"2017"')
```

Now let us *increase the number of topics to five*.

In [10]:

```
five_model = Lda(doc_term_matrix, num_topics=5, id2word=dictionary, passes=50)
print(*five_model.print_topics(num_topics=5, num_words=6), sep='\n')
```

```
(0, '0.015*"tax" + 0.012*"market" + 0.011*"u" + 0.007*"china" + 0.006*"global" + 0.005*"repatriation"')
(1, '0.011*"amazon" + 0.010*"sphere" + 0.009*"company" + 0.009*"seattle" + 0.007*"2018" + 0.006*"employee"')
(2, '0.013*"inc" + 0.009*"jan" + 0.008*"blade" + 0.007*"bucket" + 0.006*"pusher" + 0.005*"skid"')
(3, '0.017*"plant" + 0.010*"case" + 0.006*"wardian" + 0.005*"care" + 0.005*"year" + 0.005*"terrarium"')
(4, '0.009*"share" + 0.009*"market" + 0.007*"2017" + 0.007*"caterpillar" + 0.006*"estimate" + 0.006*"median"')
```

It seems that three of these topics are on Caterpillar, with the other two being on China and Amazon. The first Caterpillar topic is somewhat distinct from the other two, perhaps talking about the financial performance of the company in a given time period. The other two are more difficult to distinguish. However, when we *increase to six topics* this overlap goes away, and all topics become distinct and somewhat understandable.

In [14]:

```
six_model = Lda(doc_term_matrix, num_topics=6, id2word=dictionary, passes=50)
print(*six_model.print_topics(num_topics=6, num_words=6), sep='\n')
```

```
(0, '0.015*"sphere" + 0.014*"amazon" + 0.013*"seattle" + 0.008*"blade" + 0.008*"monday" + 0.007*"grand"')
(1, '0.011*"median" + 0.010*"estimate" + 0.009*"university" + 0.009*"2017" + 0.008*"city" + 0.007*"town"')
(2, '0.029*"market" + 0.012*"caterpillar" + 0.010*"share" + 0.010*"report" + 0.009*"industry" + 0.008*"product"')
(3, '0.007*"iot" + 0.006*"product" + 0.006*"cart" + 0.005*"industrial" + 0.005*"caterpillar" + 0.005*"manufacturing"')
(4, '0.014*"tax" + 0.010*"u" + 0.007*"jan" + 0.006*"inc" + 0.005*"would" + 0.005*"china"')
(5, '0.018*"plant" + 0.011*"case" + 0.009*"care" + 0.007*"wardian" + 0.005*"health" + 0.005*"terrarium"')
```

What happens when we increase to seven topics? Here we see there are diminishing returns as now there is more overlap between topics, especially topics indexed at 1 and 2.

In [12]:

```
seven_model = Lda(doc_term_matrix, num_topics=7, id2word=dictionary, passes=50)
print(*seven_model.print_topics(num_topics=7, num_words=6), sep='\n')
```

```
(0, '0.010*"iot" + 0.007*"equipment" + 0.007*"truck" + 0.007*"service" + 0.007*"mnubo" + 0.006*"company"')
(1, '0.018*"share" + 0.017*"jan" + 0.016*"inc" + 0.013*"caterpillar" + 0.009*"stock" + 0.009*"company"')
(2, '0.011*"tax" + 0.010*"plant" + 0.009*"u" + 0.005*"case" + 0.005*"one" + 0.005*"city"')
(3, '0.008*"house" + 0.006*"2018" + 0.006*"et" + 0.005*"like" + 0.005*"year" + 0.004*"2"')
(4, '0.011*"market" + 0.009*"blade" + 0.008*"vehicle" + 0.008*"pusher" + 0.007*"snow" + 0.006*"end"')
(5, '0.025*"market" + 0.011*"sphere" + 0.011*"amazon" + 0.010*"seattle" + 0.008*"industry" + 0.007*"2018"')
(6, '0.009*"health" + 0.008*"company" + 0.006*"new" + 0.006*"care" + 0.005*"employee" + 0.005*"cost"')
```

Explain how you selected N

So at three topics each topic is distinct, but we do not know if we are covering all topics actually discussed in the articles. At five topics, we start to see some overlap in topics. This overlap goes away when we increase to six topics, but returns when we use seven. **For this reason, I chose the six topic model.**

We can visualize the six topic model as below. There is some overlap between topics, but this appears to be incidental, and we see some distinct separation in other topics.

In [15]:

```
lda_display = pyLDAvis.gensim.prepare(six_model, doc_term_matrix, dictionary, sort_topics=False)
pyLDAvis.display(lda_display)
```

Out[15]:

Selected Topic: 0 Previous Topic Next Topic Clear Topic

Slide to adjust relevance metric:(2)
λ = 1