A Report on

NEWS ARTICLES SENTIMENT ANALYSIS AND TOPIC MODELING

Submitted By

Yashasvi Singh

Submitted To

Professor. Lu Xiao

May 8th, 22

Abstract

Sentiment analysis is one of the widely used natural language processing techniques, particularly in the classification approach for extracting the content with emotions and attitudes. In such a manner, sentiment analysis may thus be thought of as a method for quantifying qualitative data using score markers. Despite the fact that sentiments are mostly subjective, quantitative analysis of emotions has many useful practices. Because of recent breakthroughs in machine learning and deep learning, sentiment analysis algorithms have grown more efficient. It takes a long time to manually classify emotional words. In today's world, where digital data is constantly being collected. Looking for insights from the acquired data might become difficult and tedious. Topic modeling is mainly used for semantic analysis and text mining in Natural Language Processing. Topic Modeling was developed as a method for organizing, searching, and comprehending large amounts of textual mental material that derives hidden patterns in the corpus resulting in improved decision-making. The topic is a probability distribution in Topic Modeling, using all characters in the text as the supporting set, showing the frequency of characters in the topic; characters with high importance to the topic have a higher likelihood of occurring. Our group has used both sentiment analysis and topic modeling (LDA) techniques to analyze news articles so that we can know the overall emotions of the news and frequently reported topics.

Keywords: news, sentiment analysis, topic modeling, LDA

The Table of Contents for this report is listed below.

Introduction

Throughout history, news plays an important role in our daily life. People obtain information through various media and news sites and get to know topics they are interested in or notable information that they need to pay attention to. New media companies focus on delivering news to either the general public or the target public. In our project, we get data from three new sites which are Breitbart News, Fox News, and Newsmax. All three news media companies deliver their news to the same target public. Fox News is one of the biggest news media companies in the United States. Breitbart and Newsmax are smaller companies compared to Fox News. We are interested in how these news sites shift main topics over time and the similarities or differences in topic selection during the same period of time for these three news media companies since they target the same audiences.

The goals of our project are mainly divided into five parts, which are data gathering, data-preprocessing, exploratory data analysis, sentiment analysis, and topic modeling.

Dataset Gathering

We gathered data from three sources. The first source, Fox News, has data from December 29th, 2021 to March 14th, 2022. The data includes a wide range of topics in the political-sphere covered by Fox News. Breitbart, another popular conservative news site, has data from January 29th, 2022 to April 18th, 2022, and Newsmax from July 1st, 2021 to April 17th, 2022. These three sites cover a wide spectrum of conservative news-watchers.

The dataset was created mostly automatically via web-scraping methodologies. Newsmax provided a simple web page with complete archives of posted content. Using this web page would provide a list of Universal Resource Locators (URL). Breitbart uses statically paginated content. Clicking 'next page' would change the URL and show different content. By programmatically searching the paginated results, we could obtain the list of URLs.

Fox News had no simple method for getting article URLs. Fox News provided a button that would add more articles to the current page using a JavaScript AJAX call. To retrieve a list of URLs of Fox News articles, that button had to be clicked many times, until the articles reached the beginning of the date range desired (in this case, New Year 2022). Once this button had been clicked enough times, the page's source-HTML was saved and parsed using the Python BeautifulSoup library. Parsing this final HTML provided the list of URLs: all articles shown on that page.

```
def do_breitbart():
    def inner_breitbart(i):
        page = f'https://www.breitbart.com/politics/page/{i}/'
        soup = BeautifulSoup(requests.get(page).text, 'html.parser')
        alist = soup.find('section', class_="aList")
        if not alist:
            print("NO ALIST FOR " + page)
            return []
        return ['https://www.breitbart.com' + article.find('h2').find('a').get('href')
                    for article in alist.find_all('article')]
```

```
https://www.foxnews.com/politics/zelenskyy-to-address-congress-wednesday-as-war-in-ukraine-rages-on
https://www.foxnews.com/politics/ukraine-russia-biden-mig-fighter-jets-escalate
https://www.foxnews.com/politics/trump-backed-wyoming-gop-primary-challenger-hageman-urges-sending-cheney-home-to-virginia
https://www.foxnews.com/politics/lorie-smith-web-designer-free-speech
https://www.foxnews.com/politics/biden-delaware-election-law-trouble-lawsuit-early-voting-state-constitution
https://www.foxnews.com/politics/dr-oz-now-a-gop-senate-candidate-had-jazz-jennings-transgender-parents-transgender-surgeon-on-his-show
https://www.foxnews.com/politics/jim-hagedorns-widow-jennifer-carnahan-announces-run-for-his-open-house-seat
```

Once the lists of URLs had been made, web-scrapers were built to access those pages and then scrape them for their content. As described in the next section, all relevant data was gathered: The title (and sometimes a subtitle), article text, published date, and even the author. Each article

was rendered into its own JSON object. There were three files, one for each news site, and each line contained a JSON object to be used for analysis.


```
1  {"title": "Convenient Timing: EU Investigates Le Pen Ahead of French Election", "text": "\nPARIS (AP) \u2013 Paris prosecutors are studying a
   report by the European Union\u2019s fraud agency accusing French far-right presidential candidate Marine Le Pen and other members of her
   nationalist party of misusing public funds while serving in the European Parliament.\nThe report was disclosed by French investigative news site
   Mediapart days before Le Pen faces incumbent Emmanuel Macron in a runoff election Sunday that could determine Europe\u00b4s future direction. Le
   Pen\u2019s party National Rally seeks to diminish the EU\u00b4s powers.\nParty lawyer Rodolphe Bosselut said she denies wrongdoing, and questioned
   the timing of the Mediapart publication, just before the presidential runoff.\nMacron, a pro-EU centrist, leads Le Pen in polls ahead of
   Sunday\u00b4s vote, though the race is tighter than when they faced off in 2017.\nEU fraud agency OLAF submitted its report last month to the Paris
   prosecutor\u00b4s office, which is \u201cin the course of analyzing it,\u201d the prosecutor\u00b4s office said Monday. No formal investigation has
   yet been opened, and no further details were released.\nAccording to Mediapart, the OLAF report found that Le Pen, her firebrand father and party
   founder Jean-Marie Le Pen and other party members who served in the European Parliament used 617,000 euros of public money for
   \u201cfictitious\u201d reasons, notably for the benefit of companies close to the party. The fraud office is reportedly seeking reimbursement of
   the funds and potential fraud and embezzlement charges.\nOLAF accused party members of \u201cgrave violations\u201d and said the
   \u201cinappropriate behavior\u201d of members of National Rally \u2013 formerly called the National Front \u2013 \u201cimperiled the reputation of
   the Union\u00b4s institutions,\u201d according to Mediapart.\nOLAF didn\u2019t immediately respond to requests for comment Monday, a holiday in
   Belgium and several European countries.\nIt\u2019s not the first time Le Pen and her party were accused of misusing EU funds. Among several legal
   affairs that have dogged her party, Le Pen was handed preliminary charges in 2018 based on a separate investigation by OLAF accusing National Rally
   members of using aides on the European Parliament\u00b4s payroll for the party\u00b4s political activity.\nLe Pen, who served in the European
   Parliament from 2004-2017, met with supporters Monday in the Normandy town of Saint-Pierre-en-Auge. She and Macron face a crucial debate on
   Wednesday.", "date": "18 Apr 2022", "author": "Breitbart London", "url": "https://www.breitbart.com/europe/2022/04/18/
   convenient-timing-eu-fraud-agency-investigates-le-pen-ahead-of-french-election/"}
```


Each site had a different layout, so scraping code had to be specialized. This was done by visual inspection of site source code, and then utilizing BeautifulSoup to write a scraper, parsing out the content, and saving it correctly. Article content was regularly split between many HTML tags, so care had to be taken that the entire article content was retrieved.

```python
# Returns a dict containing the info of the article.
def scrape_fox(url):
    site = requests.get(url)
    soup = BeautifulSoup(site.text, features="lxml")

    source = "foxnews.com" # url[:10 + url[10:].find('/')]

    title = soup.title.string
    if title.endswith("| Fox News"):
        title = title[:-len("| Fox News")].strip()
    byline = soup.find(class_='author-byline')
    author = None
    if byline:
        author = byline.find('a').text
    else:
        return None

    text = ""
    body = soup.find(class_="article-body")
    for p in body.find_all('p', recursive=False):
        text = text + "\n" + p.text
    sub_headline_elem = soup.find(class_="sub-headline")
    if sub_headline_elem:
        text = sub_headline_elem.text + "\n" + text

    published = soup.find('time').text
    return { "title": title
           , "text": text
           , "date": published
           , "author": author
           , "url": url }
```

This process took many hours, as properly web scraping requires timing requests to not overload the server. A Denial of Service (DOS) attack is easy to accidentally create. However, it is also easy to spot one, and then the scraper would be IP-banned from accessing the websites. To be safe, we required waiting for 1 second after a page finished being scraped. Overall, that increased the scraping time dramatically. Scraping a page may take over 1 second, and then 1 more second must be spent waiting, evading detection from the sites.

Overall, over 10 hours of compute-time were needed to gather the data. However, over 50MB of data was gathered. Months of articles by these sites.

Dataset Description

After finishing collecting data, we saved the collected data into three JSON files according to the news media sites. For all the news data frames, there are five columns, which are ‘title’ (titles of the news), ‘text’ (news articles), ‘date’ (date published), ‘author’ (the writers of the news), and ‘url’ (website links of the news). There are 6000 records collected for Breitbart news, 3535 records for Fox, and 8590 records for Newsmax. The data frames of three data sets are listed below:

```
breitbart.head()
```

	title	text	date	author	url
0	Convenient Timing: EU Investigates Le Pen Ahea...	PARIS (AP) – Paris prosecutors are studying a ...	2022-04-18	Breitbart London	https://www.breitbart.com/europe/2022/04/18/co...
1	Kamala Harris Uses 'Wordle' Phone Game as a 'B...	Vice President Kamala Harris plays the popular...	2022-04-18	Charlie Spiering	https://www.breitbart.com/politics/2022/04/18/...
2	BLM Nuttery: Park May be Given Name of Gaff Pr...	Leftist politicians in one local council in Br...	2022-04-18	Peter Caddle	https://www.breitbart.com/europe/2022/04/18/bl...
3	Brexit Blunder: Migrant Worker Visas Increase ...	Boris Johnson’s post-Brexit immigration scheme...	2022-04-18	Kurt Zindulka	https://www.breitbart.com/europe/2022/04/18/bo...
4	Durham: Five Hillary Clinton Associates Are Ta...	Five associates of Hillary Clinton and her pre...	2022-04-18	Joel B. Pollak	https://www.breitbart.com/politics/2022/04/18/...

```
fox.head()
```

	title	text	date	author	url
0	Russia-Ukraine: Zelenskyy to deliver virtual a...	Ukrainian President Volodymyr Zelenskyy is exp...	2022-03-14 10:14:00	Brooke Singman	https://www.foxnews.com/politics/zelenskyy-to-...
1	Russia-Ukraine war: Congress eyes MiG-29 fight...	Biden admin worries about potential escalation...	2022-03-14 09:55:00	Tyler Olson	https://www.foxnews.com/politics/ukraine-russi...
2	Trump-backed Wyoming GOP primary challenger Ha...	Hageman spotlights Cheney's roots in Northern ...	2022-03-14 08:10:00	Paul Steinhauser	https://www.foxnews.com/politics/trump-backed-...
3	Web designer who refuses to create sites for s...	Lorie Smith says Colorado is violating her Fir...	2022-03-14 07:33:00	Ronn Blitzer	https://www.foxnews.com/politics/lorie-smith-w...
4	Biden's home state election law trouble: Lawsu...	Delaware resident says law that takes effect t...	2022-03-14 06:19:00	Fred Lucas	https://www.foxnews.com/politics/biden-delawar...

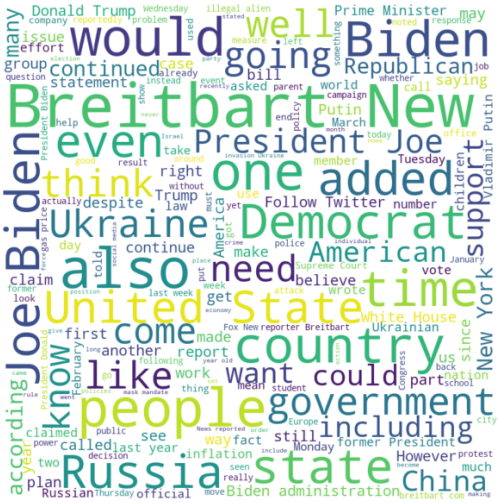
```
newsmax.head()
```

	title	text	date	author	url
0	Panetta: Send Ukraine Every Weapon We Can	Former United States Secretary of Defense Leon...	2022-04-17 21:26:00	Nick Koutsobinas	https://www.newsmax.com/politics/leon-panetta-...
1	Former Head of US Army in Europe: $800M in Aid...	The former head of the U.S. Army in Europe tol...	2022-04-17 18:45:00	Nick Koutsobinas	https://www.newsmax.com/politics/russia-ukrain...
2	Some Skeptical Biden's Student-Debt Action Wil...	The White House earlier this week said canceli...	2022-04-17 16:00:00	Eric Mack	https://www.newsmax.com/politics/student-loan-...
3	Catholic Charities Nun Decries Traffickers Tak...	The executive director of the Catholic Chariti...	2022-04-17 13:53:00	Fran Beyer	https://www.newsmax.com/politics/catholic-char...
4	COVID Coordinator: Latest Variant Surge Won't ...	The White House COVID coordinator, Dr. Ashish ...	2022-04-17 13:29:00	Fran Beyer	https://www.newsmax.com/politics/ashish-jha-va...

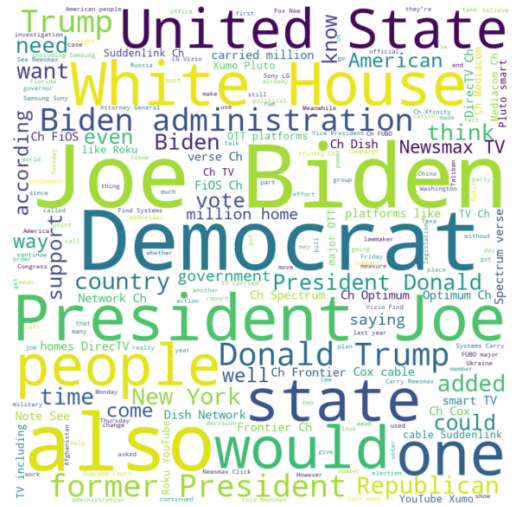
Exploratory Data Analysis

To understand the news content for each news site, down below, we listed word clouds for all three news sites using the words in the text column.

Breitbart

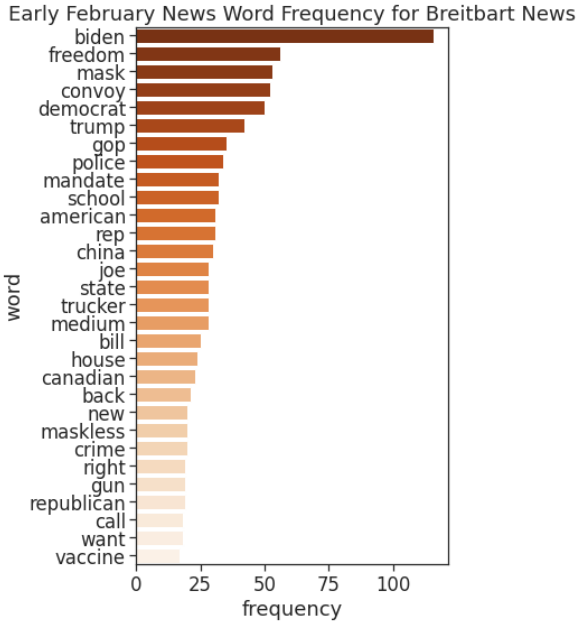


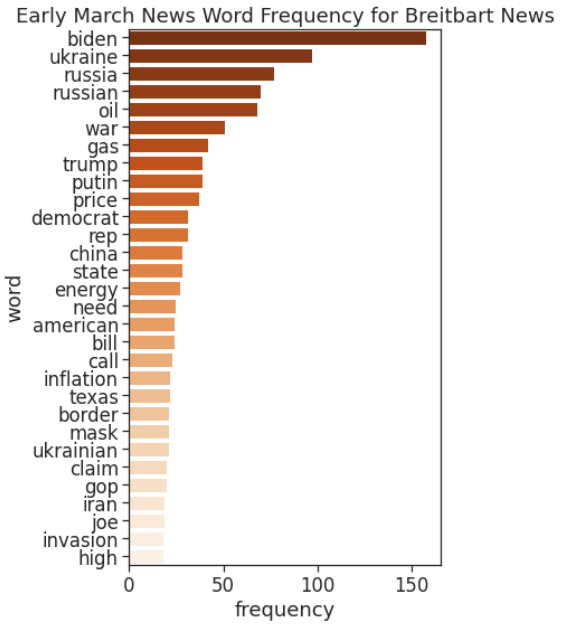
Fox News

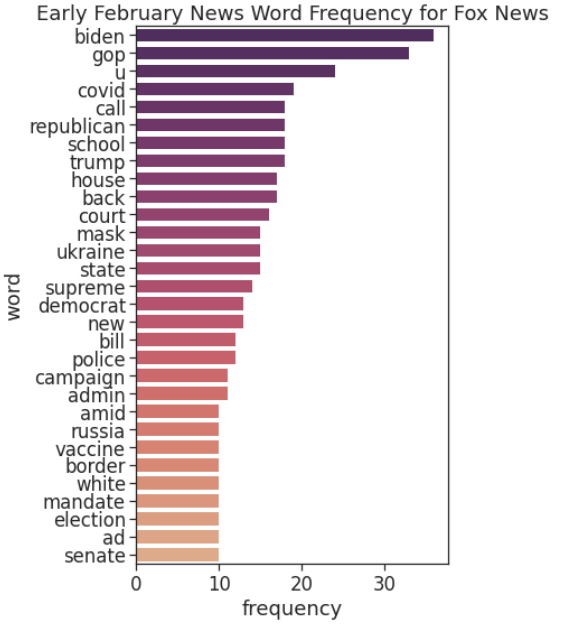


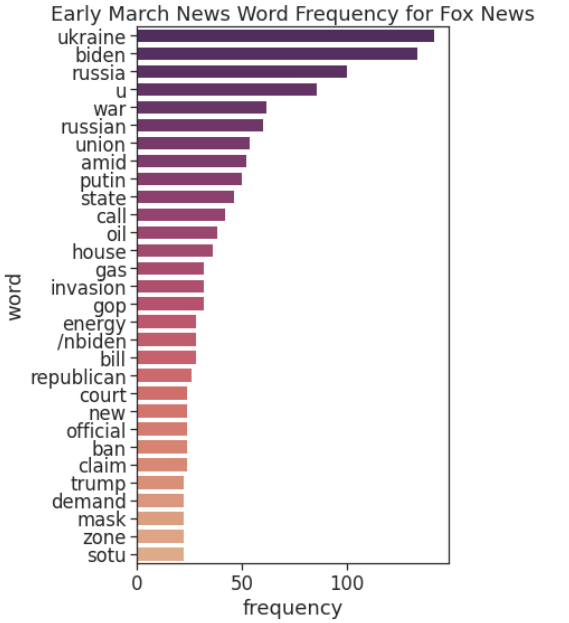
Newsmax

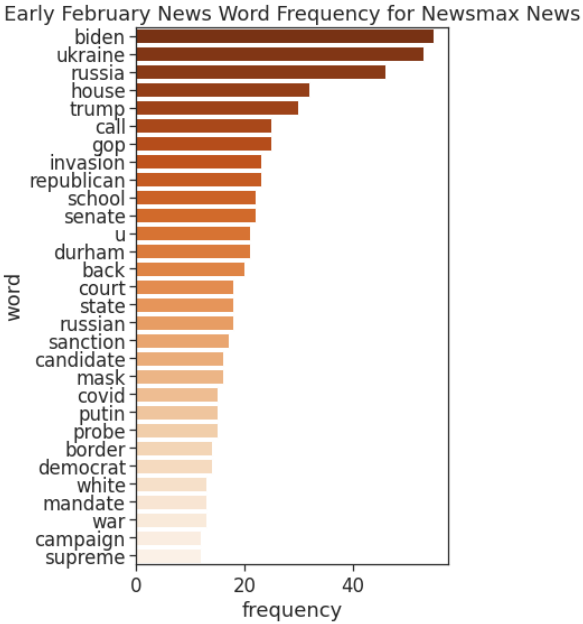
These word clouds can only give us some insights on what are the topics these news sites' articles are trying to cover, but news articles usually are newly received or notable information. The information we obtain from the word clouds above covers a long period of time. Therefore, breaking down the news into different time periods might help us better understand each news media's way of selecting topics to report.

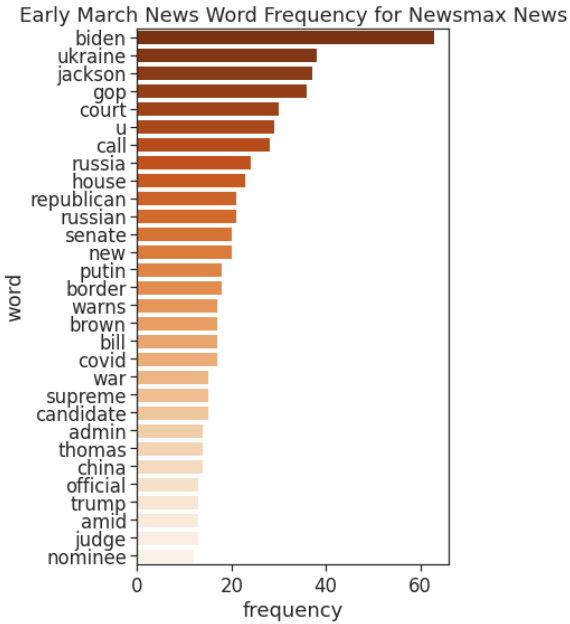

Early February News Word Frequency for Breitbart News
biden
freedom
mask
convoy
democrat
trump
gop
police
mandate
school
american
rep
china
joe
state
trucker
medium
bill
house
canadian
back
new
maskless
crime
right
gun
republican
call
want
vaccine
word
0
25
50
75
100
frequency


Early March News Word Frequency for Breitbart News
biden
ukraine
russia
russian
oil
war
gas
trump
putin
price
democrat
rep
china
state
energy
need
american
bill
call
inflation
texas
border
mask
ukrainian
claim
gop
iran
joe
invasion
high
word
0
50
100
150
frequency


Early February News Word Frequency for Fox News
biden
gop
u
covid
call
republican
school
trump
house
back
court
mask
ukraine
state
supreme
democrat
new
bill
police
campaign
admin
amid
russia
vaccine
border
white
mandate
election
ad
senate
word
0
10
20
30
frequency


Early March News Word Frequency for Fox News
ukraine
biden
russia
u
war
russian
union
amid
putin
state
call
oil
house
gas
invasion
gop
energy
/nbiden
bill
republican
court
new
official
ban
claim
trump
demand
mask
zone
sotu
word
0
50
100
frequency


Early February News Word Frequency for Newsmax News
biden
ukraine
russia
house
trump
call
gop
invasion
republican
school
senate
u
durham
back
court
state
russian
sanction
candidate
mask
covid
putin
probe
border
democrat
white
mandate
war
campaign
supreme
word
0
20
40
frequency


Early March News Word Frequency for Newsmax News
biden
ukraine
jackson
gop
court
u
call
russia
house
republican
russian
senate
new
putin
border
warns
brown
bill
covid
war
supreme
candidate
admin
thomas
china
official
trump
amid
judge
nominee
word
0
20
40
60
frequency

We selected two time periods: February 1st to 14th and March 1st to 14th.
Early February

- Breitbart
- Fox
- Newsmax

From the result, we can see that the three news sites talk most frequently about Biden. It reflects the commonality of the topics discussed by the three news sites. What is more, We can also see the commonality between the two news sites. Like Breitbart News and Fox News, they often use words like “mask, maskless, covid”. We can roughly estimate that the topic they are talking about is related to the current epidemic situation. For Newsmax News, we can infer from the data that they are talking more about international news, because we can see some countries' names, like Ukraine and Russia. We know that war broke out between Russia and Ukraine at the end of February, but the data we collected came from the beginning of February. Compared to the other two news media companies the reports of the Newsmax seems forward-looking.

Early March

- Breitbart
- Fox
- Newsmax

At the beginning of March, the war between Russia and Ukraine broke out in an all-around way. We can see that these three news websites are talking about the same topics, such as “Putin, war, Russian, and Ukraine”. We can also see that some words appear on these three news websites, such as oil and price, which shows that the war between Russia and Ukraine has brought great changes to the oil price. Here we take the comparative method instead of analyzing the three news websites separately. This method will be more meaningful and get more information. Through these three sets of data, we can see the characteristics of news topics, real-time, forward-looking, and commonality.

Sentiment Analysis

Sentiment analysis algorithms have become more efficient due to recent advances in machine learning and deep learning. The manual classification of sentiment words takes a long time. To automate the sentiment analysis process, there are two approaches that are commonly used.

- *Rule-Based* - It matches a collection of words in a text against a word dictionary with key phrases to find polarity.
- *Automatic* - Employs machine learning methods, there is no need to preprocess data or train a classifier.

As for how to determine the sentiment polarity, we used the textblob library to perform the labeling.

```
from textblob import TextBlob
from textblob.sentiments import NaiveBayesAnalyzer
```

```
breitbart['polarity'] = ''
for i,x in breitbart.text.iteritems():
    label = TextBlob(x)
    breitbart['polarity'][i] = label.sentiment.polarity
```

```
[ ]  def polarity_to_label(x):
         if(x <= 0):
             return 'negative'
         if(x > 0):
             return 'positive'
     breitbart.label = breitbart.polarity.apply(polarity_to_label)
```

```
[ ]  breitbart.label.value_counts()
```

```
pos    4571
neg    1429
Name: label, dtype: int64
```

```
[ ] newsmax.label.value_counts()
```

```
positive    7301
negative    1289
Name: label, dtype: int64
```

```
[ ] newsmax.to_csv("newsmax.csv", encoding='utf-8')
    files.download('newsmax.csv') #download the csv file
```

```
[ ] fox.label.value_counts()
```

```
positive    2900
negative     635
Name: label, dtype: int64
```

These three sets of data more clearly reflect the attitude of news websites towards using words. The positive contents outweigh the negative ones.

Then, we want to build a binary text classifier to classify the sentiment of text using the news data. Using the Fox News annotated dataset, we can build our own model.

```
df[['text','label']].head()
```

	text	label
0	Ukrainian President Volodymyr Zelenskyy is exp...	pos
1	Biden admin worries about potential escalation...	pos
2	Hageman spotlights Cheney's roots in Northern ...	pos
3	Lorie Smith says Colorado is violating her Fir...	pos
4	Delaware resident says law that takes effect t...	pos

```
sentiment_label = df.label.factorize()
sentiment_label # pos is 0, neg is 1
```

```
(array([0, 0, 0, ..., 0, 0, 0]), Index(['pos', 'neg'], dtype='object'))
```

After reading the data into a data frame, we factorize the sentiment labels since the machine can only understand the numeric data.

```
tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(news)
```

```
encoded_docs = tokenizer.texts_to_sequences(news) # Replace the words with their assigned numbers using the text_to_sequence()
```

Then, we tokenized text and used texts to sequences to assign numbers to the texts.

```
padded_sequence = pad_sequences(encoded_docs, maxlen=200)
padded_sequence # Use padding to pad the sentences to have equal length.
```

```
array([[ 276,  450,   22, ...,   22,  109,   53],
       [  12,    1,  404, ...,   22,  109,   53],
       [  13,  248,  181, ...,   10, 1144, 1709],
       ...,
       [   1,  122,   76, ..., 2037,   82,  996],
       [  12,    6, 3414, ...,   43,  544,  345],
       [  38,  501,    6, ...,  521,   11,   53]], dtype=int32)
```

After that, we used padding to pad the sentences so that they can have equal length.

```
vocab_size = len(tokenizer.word_index) + 1
vocab_size
```

```
44638
```

Define the vocabulary size.

```
# Build the text classifier
embedding_vector_length = 64
model = Sequential()
model.add(Embedding(vocab_size, embedding_vector_length, input_length=200))
model.add(SpatialDropout1D(0.25))
model.add(LSTM(50, dropout=0.5, recurrent_dropout=0.5))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam', metrics=['accuracy'])
print(model.summary())
```

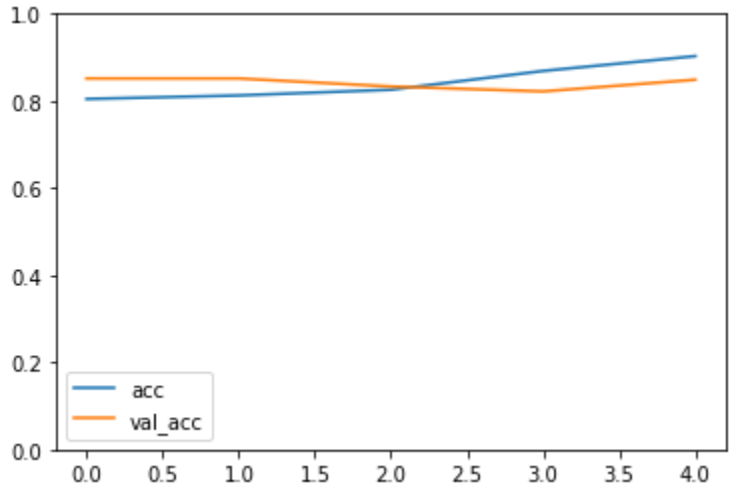
```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 200, 64)           2856832

 spatial_dropout1d_1 (Spatia  (None, 200, 64)          0
 lDropout1D)

 lstm_1 (LSTM)               (None, 50)                23000

 dropout_1 (Dropout)         (None, 50)                0

 dense_1 (Dense)             (None, 1)                 51

=================================================================
Total params: 2,879,883
Trainable params: 2,879,883
Non-trainable params: 0
_________________________________________________________________
None
```

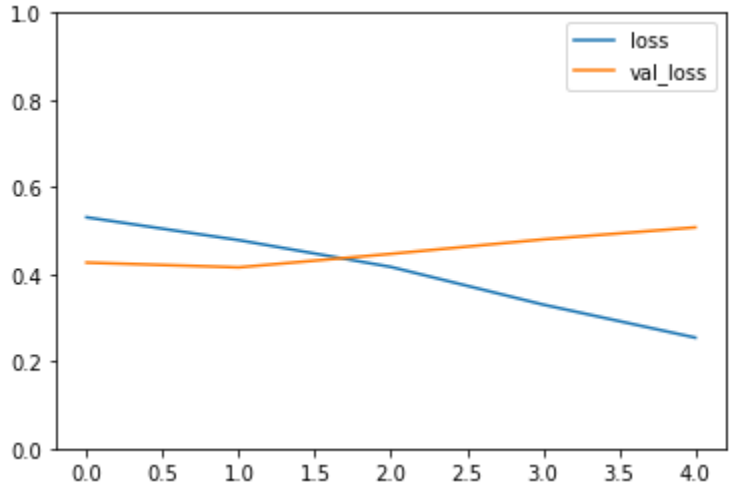
Build our text classifier.

```
history = model.fit(padded_sequence,sentiment_label[0],validation_split=0.2, epochs=5, batch_size=32) # Train the model
```

```
Epoch 1/5
89/89 [==============================] - 24s 236ms/step - loss: 0.5306 - accuracy: 0.8045 - val_loss: 0.4266 - val_accuracy: 0.8515
Epoch 2/5
89/89 [==============================] - 21s 235ms/step - loss: 0.4782 - accuracy: 0.8126 - val_loss: 0.4157 - val_accuracy: 0.8515
Epoch 3/5
89/89 [==============================] - 21s 232ms/step - loss: 0.4167 - accuracy: 0.8257 - val_loss: 0.4467 - val_accuracy: 0.8331
Epoch 4/5
89/89 [==============================] - 22s 243ms/step - loss: 0.3302 - accuracy: 0.8688 - val_loss: 0.4799 - val_accuracy: 0.8218
Epoch 5/5
89/89 [==============================] - 22s 253ms/step - loss: 0.2544 - accuracy: 0.9028 - val_loss: 0.5074 - val_accuracy: 0.8487
```

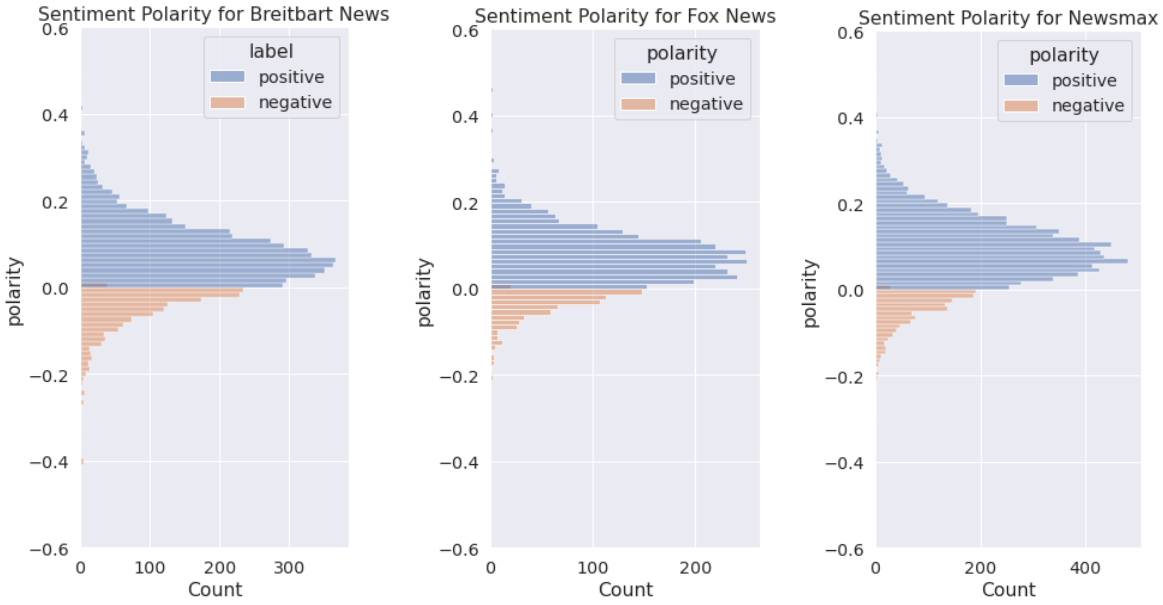
Train the sentiment analysis model.





Visualize the train (acc, loss), test (val_acc, val_loss) accuracies, and their losses. The overall accuracy is satisfying. As for the losses, more errors refer to higher losses, the model did not do a good job if high losses happen. Training loss is much lower than validation loss is also something we do not want to see because it means that the network might be overfitting. In our model, there might be some overfitting issues even after adjusting some hyperparameters, which means the model will not work as great when we use it on other datasets.

Interpretation



These are the sentiment polarities of three news sites. From the data obtained, the overall positivity in contents outweighs the negativity, this is their identical characteristic. Compared to Fox News and Newsmax, Breitbart has a slightly higher percentage of negative content. For Breitbart News and News max, the range of sentiment polarity is around 0.6. But the range of sentiment polarity for Fox News is around 0.4. It is worth mentioning again that Fox News is one of the biggest news media companies. Breitbart and Newsmax are relatively smaller companies. The wide sentiment polarity interval for Newsmax and Breitbart means that they tend to report news using more expressive words whereas Fox News is more prone to use neutral or formal words while reporting.

Topic Modeling

Using a probabilistic model, topic modeling discovers abstract topics that appear in a corpus of documents. It is commonly used as a text mining tool to uncover semantic structures within a body of text.
All topic models are built on a basic assumption:
each document is made up of a variety of topics,
each of which is made up of a collection of words.

We tried to extract topics using the Latent Dirichlet Allocation (LDA) technique on fox news data. LDA is a popular Bayesian form of pLSA that's straightforward to use. For the document-topic and word-topic distributions, it uses dirichlet priors, which allows for more generalization. Since LDA may easily generalize to new documents, it is typically more successful than pLSA.

```
data_text = df[['text']]
data_text['index'] = data_text.index
documents = data_text
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
```

```
def lemmatize_stemming(text):
    return WordNetLemmatizer().lemmatize(text, pos='v')
def preprocess(text):
    result = []
    for token in gensim.utils.simple_preprocess(text):
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3:
            result.append(lemmatize_stemming(token))
    return result
```

Lemmatizing text (Fox News articles)

```
doc_sample = documents[documents['index'] == 3534].values[0][0]
print('original document: ')
words = []
for word in doc_sample.split(' '):
    words.append(word)
print(words)
print('\n\n tokenized and lemmatized document: ')
print(preprocess(doc_sample))
```

```
original document:
['Progressives', 'failed', 'on', 'their', 'biggest', 'policy', 'goals', 'b


 tokenized and lemmatized document:
['progressives', 'fail', 'biggest', 'policy', 'goals', 'show', 'leverage',
```

Let's preview a document. There are differences between original and lemmatized texts.

```
dictionary = gensim.corpora.Dictionary(processed_docs)
count = 0
for k, v in dictionary.iteritems():
    print(k, v)
    count += 1
    if count > 10:
        break
```

```
0 action
1 additional
2 address
3 admiration
4 aggression
5 alliance
6 ally
7 amid
8 announce
9 artery
10 article
```

Create a dictionary

```
dictionary.filter_extremes(no_below=15, no_above=0.5, keep_n=100000)
```

Filtering out texts that exist in less than 15 articles or more than 0.5 articles. Then, keep the first 100000 most frequent tokens.

```
bow_corpus = [dictionary.doc2bow(doc) for doc in processed_docs]
bow_corpus[3534]
```

For each article, we create a dictionary reporting how many words and times those words appear.

```
from gensim import corpora, models
tfidf = models.TfidfModel(bow_corpus)
corpus_tfidf = tfidf[bow_corpus]
from pprint import pprint
for doc in corpus_tfidf:
    pprint(doc)
    break
```

```
 (36, 0.04148732789145375),
 (37, 0.04793384921043681),
 (38, 0.11095314388139911),
 (39, 0.04734096103930822),
 (40, 0.0687591678937217),
```

Apply transformation. Getting scores.

```
lda_model = gensim.models.LdaMulticore(bow_corpus, num_topics=10, id2word=dictionary, passes=2, workers=2)
```

Train the LDA model using gensim.models.LdaMulticore.

```
for idx, topic in lda_model.print_topics(-1):
    print('Topic: {} \nWords: {}'.format(idx, topic))
```

Interpretation

Explored the words occurring in that topic and their relative weight.
The top 10 topics are listed below:

```
Topic: 0
Words: 0.009*"border" + 0.008*"administration" + 0.007*"school" +
0.006*"trump" + 0.006*"covid" + 0.005*"mask" + 0.004*"year" +
0.004*"texas" + 0.004*"mandate" + 0.004*"migrants"
Topic: 1
Words: 0.022*"ukraine" + 0.018*"russia" + 0.012*"russian" + 0.009*"nato" +
0.008*"putin" + 0.005*"zelenskyy" + 0.005*"military" + 0.005*"unite" +
0.005*"ukrainian" + 0.005*"force"
Topic: 2
Words: 0.021*"russia" + 0.017*"ukraine" + 0.011*"russian" + 0.009*"putin"
+ 0.008*"trump" + 0.005*"sanction" + 0.005*"administration" +
0.005*"invasion" + 0.004*"continue" + 0.004*"ukrainian"
Topic: 3
Words: 0.015*"ukraine" + 0.009*"putin" + 0.009*"russia" + 0.007*"russian"
+ 0.006*"administration" + 0.005*"trump" + 0.004*"go" + 0.004*"court" +
0.004*"york" + 0.004*"support"
Topic: 4
```

```
Words: 0.012*"court" + 0.008*"jackson" + 0.007*"senate" +
0.006*"republican" + 0.006*"supreme" + 0.005*"trump" + 0.005*"campaign" +
0.004*"judge" + 0.004*"china" + 0.004*"white"
Topic: 5
Words: 0.006*"test" + 0.005*"border" + 0.005*"school" + 0.005*"year" +
0.005*"administration" + 0.005*"mask" + 0.004*"trump" + 0.004*"covid" +
0.004*"continue" + 0.003*"go"
Topic: 6
Words: 0.014*"vote" + 0.011*"senate" + 0.009*"democrats" + 0.006*"police"
+ 0.006*"republicans" + 0.005*"right" + 0.005*"year" + 0.005*"republican"
+ 0.005*"election" + 0.005*"filibuster"
Topic: 7
Words: 0.008*"trump" + 0.005*"campaign" + 0.004*"russia" + 0.004*"support"
+ 0.004*"right" + 0.004*"vote" + 0.004*"federal" + 0.004*"texas" +
0.004*"year" + 0.004*"committee"
Topic: 8
Words: 0.007*"administration" + 0.007*"trump" + 0.006*"ukraine" +
0.006*"year" + 0.005*"democrats" + 0.005*"republican" + 0.005*"iran" +
0.004*"republicans" + 0.004*"russia" + 0.004*"nuclear"
Topic: 9
Words: 0.008*"ukraine" + 0.008*"trump" + 0.006*"court" + 0.006*"russia" +
0.005*"republican" + 0.005*"border" + 0.004*"go" + 0.004*"support" +
0.004*"administration" + 0.004*"democrats"
```

Word frequencies only tell us how often certain words appear, LDA topic modeling allocates words into topics, so we can gain a deeper understanding of the texts. For example, topic 0 has words such as “border”, “covid”, “trump”, and “mask”. It tells us that the news is about the things that happen inside America. However, in topic 1, we can see words like “russia”, “ukraine”, ”nato”, and “military”. These words informed us that this topic is about the Russia and Ukraine conflicts.

Conclusion

In today's world, where digital data is continually accumulating. The process of extracting useful insights from obtained data may become challenging and time-consuming. With this in mind, we concentrated on sophisticated methods such as sentiment analysis and topic modeling to accomplish such tasks automatically and efficiently. We examined sentiment analysis of news items using datasets from Newsmax, Breitbart, and Fox News. These news covers events that demonstrate emotions like positive, negative, or neutral. Sentiment analysis is used to examine human emotions found in textual data. For sentiment analysis of news stories, this project employs a lexicon-based approach. Topic modeling is also used to cluster similar words in the news collection. The extracted topics are used to gain relevant insights from the text data. We aim to showcase a representation of how news topics have changed over a period of time. There are numerous areas for exploration in sentiment analysis and topic modeling. We have focused on lexicon-based sentiment analysis in our work; however, machine learning approaches may result in more efficient solutions in future work.

References

- Vidiyala, R. (2021, June 8). Topic Modelling on NYT articles using Gensim,LDA. Retrieved from https://towardsdatascience.com/topic-modelling-on-nyt-articles-using-gensim-lda-37caa2796cd9
- Chawla, R. (2017, July 30). Topic Modeling with LDA and NMF on the ABC News Headlines dataset. Retrieved from https://medium.com/ml2vec/topic-modeling-is-an-unsupervised-learning-approach-to-clustering-documents-to-discover-topics-fdfbf30e27df
- S. Taj, B. B. Shaikh and A. Fatemah Meghji, "Sentiment Analysis of News Articles: A Lexicon based Approach," 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), 2019, pp. 1-5, https://doi.org/10.1109/ICOMET.2019.8673428.
- Majumder, P. (2021, November 29). Web Scraping a News Article and performing Sentiment Analysis using NLP. Retrieved from https://www.analyticsvidhya.com/blog/2021/11/web-scraping-a-news-article-and-performing-sentiment-analysis-using-nlp/
- S. Taj, B. B. Shaikh and A. Fatemah Meghji, "Sentiment Analysis of News Articles: A Lexicon based Approach," 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), 2019, pp. 1-5, https://doi.org/10.1109/ICOMET.2019.8673428.
- Dataset Resources -
  https://www.breitbart.com/
  https://www.foxnews.com/
  https://www.newsmax.com/