Capstone Project Milestone Report: *Sentiment Analysis and Topic Modeling of Political Twitter Data*

Introduction

Problem
In the 2016 United States presidential election, many opinion polls failed to predict the outcome of the election accurately. With the advent of social media, we now have many more tools at our disposal to gauge the public sentiment around polarizing topics such as politics. Through sentiment analysis of publicly available data such as Twitter data and additional tools such as topic modeling, we can better assess the pulse of the nation surrounding political events such as debates, campaigns, and elections. These tools can be used alongside polling data to develop more accurate predictions about election outcomes.

Client
A political sentiment analyzer would be of use to campaign managers, political candidates, and news outlets - all of whom would benefit with a more accurate assessment and prediction of public sentiment around high-profile political events in the public eye. Such a tool can help better plan the direction of campaigns by correctly identifying the sentiment around their political ideologies. Furthermore, understanding the topics being discussed by their constituents on social media can help guide political candidates on which topics to focus on at campaign rallies and debates.

Dataset
~86,000 tweets were extracted using the Twitter API that mentioned the 3 most popular current presidential candidates from the Democratic party: Bernie Sanders (~35,000 tweets) , Elizabeth Warren (~26,000 tweets) , and Vice President Joe Biden (~25,000 tweets).

Approach
With the upcoming 2020 presidential election, our aim is to develop a sentiment analyzer that can predict the sentiments of tweets surrounding the Sanders, Warren, and Biden. The sentiment will be divided into 3 categories: positive, negative, neutral. In addition, we would like to do topic extraction on the tweets to better understand the topics being discussed surrounding these candidates. For the sentiment analysis, we first need to label the unlabeled Twitter data. In order to do this efficiently, we will use lexicon-based tools (VADER, TextBlob, AFINN) to label the tweets into the 3 sentiment categories. We will then use supervised ML (classification) and deep learning algorithms to develop a sentiment analysis predictor that can predict the sentiment of new tweets. For the topic modeling, we will use unsupervised learning (Latent

Dirichlet Allocation, Non-negative Matrix Factorization etc.) to find patterns in the tweets and cluster words into topics that will be manually assessed and labeled.

Data Cleaning

Dropping Duplicate Tweets

The tweets for Sanders, Warren, and Biden were all extracted separately through Tweepy, which uses Twitter's API. The extended tag was used to get the full text of tweets and re-tweets were filtered out to avoid as many duplicates as possible.

After assembling all ~86,000 tweets in a pandas dataframe, duplicate tweets were removed based on their Twitter ID, as well as based on the full text. This removed ~5,000 tweets, leaving us with 81633 tweets. Twitter's API also returns a lot of extraneous data - only the tweet ID and text columns were retained. Each tweet was also tagged as "sanders", "warren", or "biden" to easily identify which candidate's name was used to obtain the tweet.

Cleaning Tweets

In order to label the tweets using lexicon-based methods and eventually perform feature extraction for machine learning, the tweets first need to be cleaned. The text in an uncleaned tweet contains a lot of "noise" - characters and objects that do not add any helpful information in sentiment analysis and can negatively affect the performance of a classifier. To clean the tweets, the following steps were taken prior to labeling:

- Removing HTML tags using BeautifulSoup (*e.g. &amp, &quot etc.*)
- Removing @mentions
- Removing URLs
- Expanding Contractions (*"don't" to "do not", "I'm" to "I am" etc.*)
- Removing special characters (punctuation, numbers, # before #hashtags etc.)
- Removing extra white space and new line breaks

Examples of some cleaned tweets are shown on the next page.

```
df.full_text[5]
```

```
'***NEW KOS POLL***\n\nAfter all the trash Markos has been tweeting about #BernieSanders lately, we need to make sure
#Bernie wins this one.\n\nGo VOTE for #Bernie2020 and then retweet to spread the word!\n\n#NotMeUS\n#MedicareForAll\n
\nhttps://t.co/Vw08gMFkRt'
```

```
df.clean_tweets[5]
```

```
'NEW KOS POLL After all the trash Markos has been tweeting about BernieSanders lately we need to make sure Bernie win
s this one Go VOTE for Bernie and then retweet to spread the word NotMeUS MedicareForAll '
```

```
df.full_text[3]
```

```
'Schadenfreude: tfw you sit back and watch Bernie Sanders campaign totally unravel (while Warren's soars) after he sa
id he would have beaten Trump in 2016 had he been the nominee. \n\nI will 💯 vote for whichever Dem wins the primary,
but I'm enjoying this a little bit...'
```

```
df.clean_tweets[3]
```

```
'Schadenfreude tfw you sit back and watch Bernie Sanders campaign totally unravel while Warrens soars after he said h
e would have beaten Trump in had he been the nominee I will vote for whichever Dem wins the primary but I am enjoying
this a little bit '
```

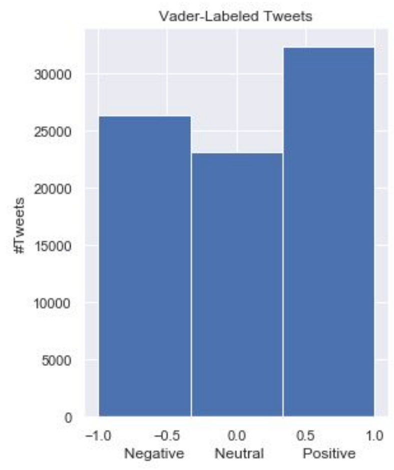
Data Labeling

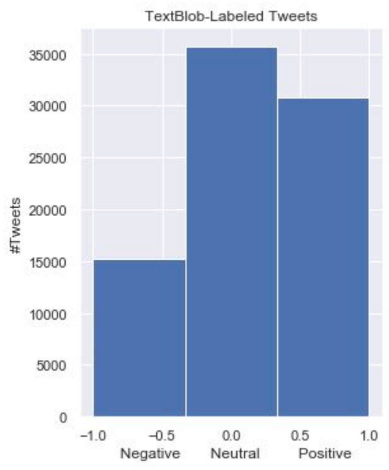
In order to perform supervised machine learning, we need labeled data. Physically going through and labeling thousands of tweets is a very time-consuming task and impractical for large datasets. Instead, we can use lexicon-based methods that use dictionaries to calculate a polarity score based on the words used in a document (i.e. tweet). The polarity score can range based on the tool but typically negative scores indicate a negative sentiment, positive scores indicate a positive sentiment, and scores close to zero imply a neutral or no sentiment. There are many lexicon-based tools out there.

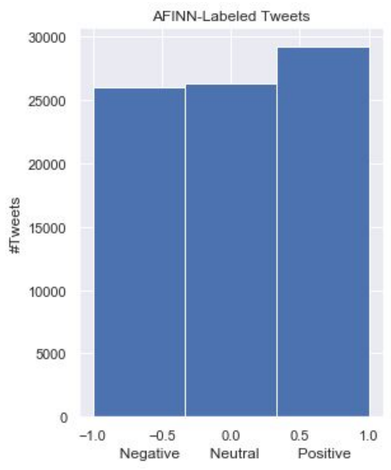
We will use 3 of them in this project: Vader, TextBlob, and AFINN. Vader and TextBlob both return a score between -1 and 1 while AFINN returns a score between -5 and 5. AFINN's output was therefore was normalized to yield the same range as VADER and Textblob. For each tweet, 3 different polarity scores were calculated using each of the 3 methods and a "positive", "negative", or "neutral" sentiment was assigned based on the following criteria, obtained from Vader's documentation:

Polarity Score	Sentiment
-1 to -0.05	Negative
-0.05 to 0.05	Neutral
0.05 to 1	Positive

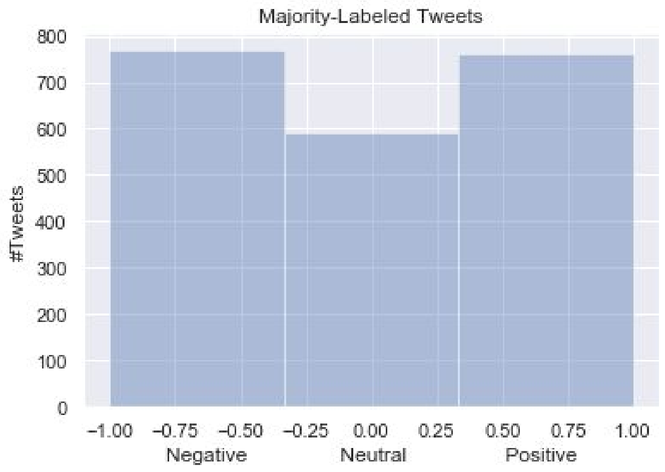
The sentiment outputs from the 3 sentiment analyzers are shown in the figure below. In general, the results from VADER and AFINN agree more closely with each other while TextBlob tags many more tweets as neutral instead of negative.







We can also calculate the sentiment of a tweet as the majority sentiment of all 3 analyzers (i.e. for each tweet, assume the sentiment to be the sentiment that at least 2 of 3 analyzers predict). If all 3 analyzers output a different sentiment, we use the VADER output since published literature has shown VADER to be one of the more effective lexicon-based tools for Twitter data.



We now have 4 different columns as the sentiment label for each tweet: VADER, TextBlob, AFINN, and the Majority Label. How can we determine which sentiment is the most accurate? In order to make an informed decision, we manually labeled ~2000 tweets and cross-referenced these manual labels with the lexicon labels to identify which method resulted in the greatest overlap. We found that the majority label agreed best, with ~50% of labels matching the manually labeled sentiment.

Additional Pre-Processing

Before proceeding with converting the text to numerical features, there are a couple of additional steps we can take to reduce feature dimensionality:

- Lemmatization
- Removing stopwords

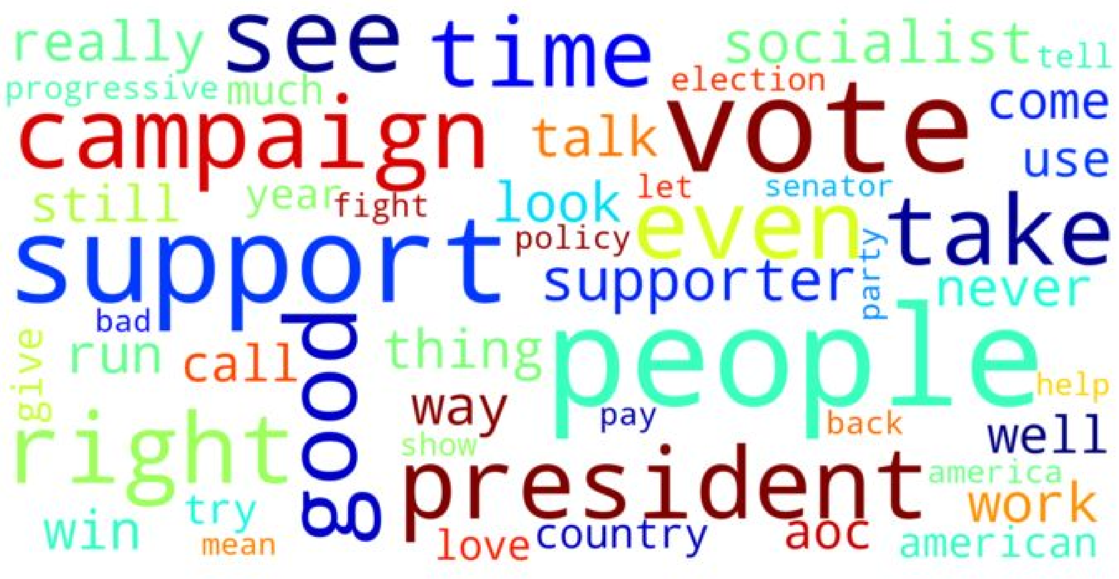
Lemmatization was used to reduce words to their root while still maintaining the meaning of the word (as opposed to stemming which often results in words that don't mean anything). A list of stopwords provided by the NLTK library was used to remove stopwords (except for negations such as no, not, but) that don't significantly contribute to the meaning of a tweet. After these pro-processing steps, any empty tweets were dropped from the corpus.

Exploratory Data Analysis

Word Clouds

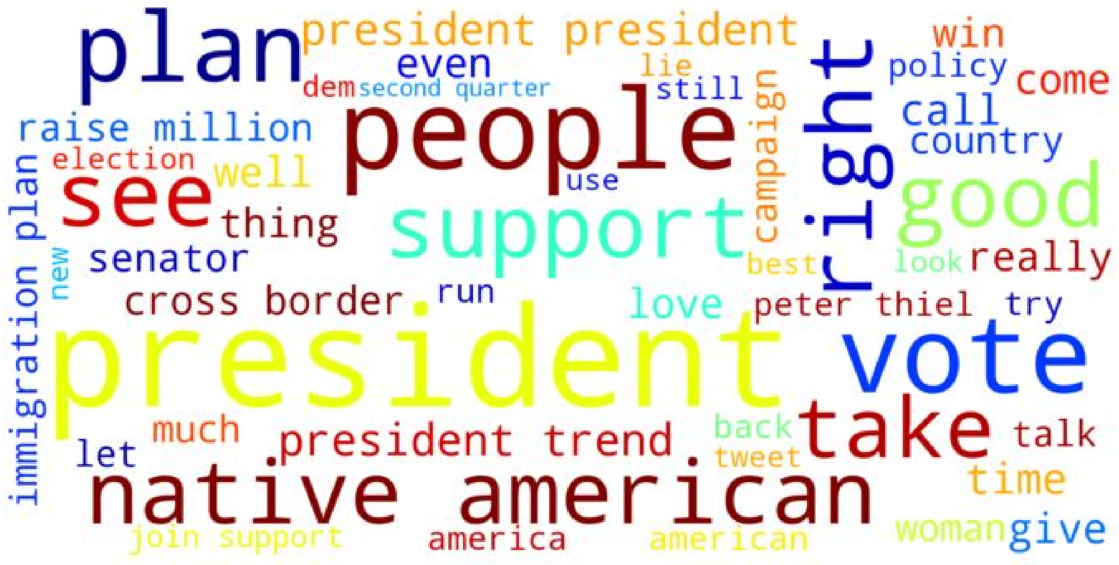
Word clouds can be an effective, visual way to explore textual data at a high level. We removed additional stopwords (names, 'say', 'think', 'want' etc.) to better visualize topics that were being discussed within the Twitter community. Below are the word clouds generated for Bernie Sanders, Elizabeth Warren, and Joe Biden, respectively.

Sanders:



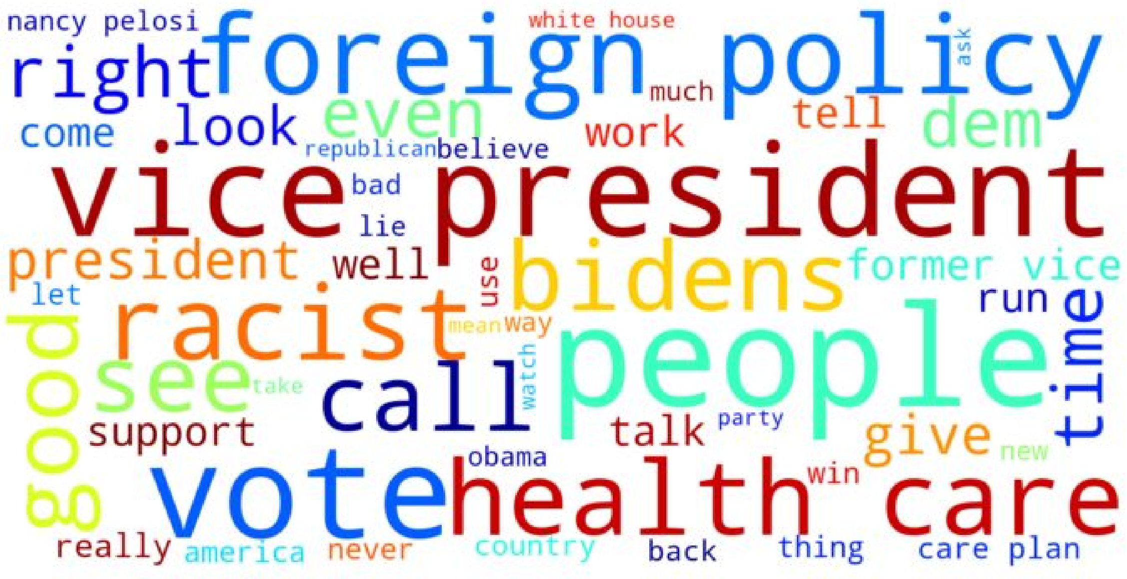
The word cloud for Sanders has positive words such as "support", "good", "right" etc.. People are talking about his campaign and, unsurprisingly, the conversations mention words like "vote" and "president". He is also recognized as a socialist but any specific topics are hard to deduce from the word cloud alone.

Warren:



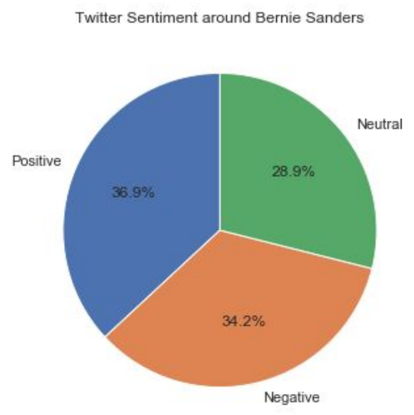
The word cloud for Warren mentions "president" and "president trend" predominantly which makes sense since "#PresidentWarren" was trending on Twitter around the time these tweets were extracted. The word "native american" is also mentioned, likely in a negative context, based on Warren's claim of being part Native American that was ridiculed heavily in the media. "Border", "cross", and "immigration plan" indicate that people are talking about her immigration and border policies.

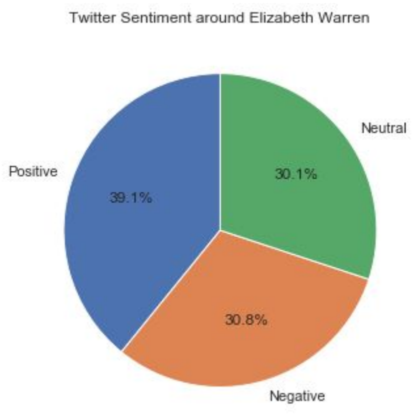
Biden:

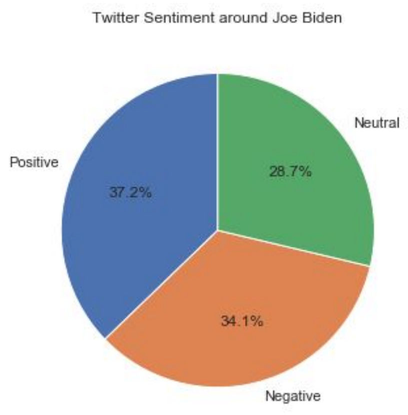


Biden's word cloud is quite informative with topics such as "foreign policy" and "health care" featured predominantly in the image. It is likely that these topics are being discussed frequently with respect to Biden. The tweets are also negatively referring to Biden as a “racist” but also as a “vice president”, which may be in a more positive, respectful context.

We can also look at the distribution of positive, negative, and neutral tweets for each candidate to see if any obvious trends emerge describing the popularity of the candidates.







Based on the lexicon-based labeling, the distribution is fairly similar with all candidates having ~36-39% positive tweets, ~30-34% negative tweets and 28-30% neutral tweets. Elizabeth Warren appears to have a slightly higher percentage of positive tweets but given the less-than-robust accuracy of the lexicon-based methods, there is likely a high standard of error among these percentage. In order to improve the accuracy of the sentiment analysis, a machine learning model should be utilized. While lexicon-based methods are static and dictionary-based, an ML model can learn relationships from the data. It can also be deployed to incorporate a feedback loop from the user which should result in higher accuracy in predictions over time. While the deployment of the model is outside the scope of this project, it is a reasonable justification for the development of a machine learning model to build a more robust sentiment analyzer.

Feature Extraction

All the pre-processed tweets were split into a training set (70%) and a test set (30%). There are several methods in Python to perform text feature extraction. 2 popular models, available in Scikit-Learn, include Bag-of-Words (BOW) and Term Frequency-Inverse Document Frequency (TF-IDF). BOW is a simple model that builds a dictionary using all tokenized words in a corpus (set of tweets) and measures the presence of known words in a document. The model is only concerned with whether known words occur in the document and how often, but does not retain any information about the order or structure of those words (hence the name “bag-of-words”).

There are some potential problems which might arise with the BOW model when it is used on large corpora. Since the feature vectors are based on absolute term frequencies, there might be some terms which occur frequently across all documents and these may tend to overshadow other terms in the feature set. The TF-IDF model tries to combat this issue by using a scaling or normalizing factor in its computation.

For this project, we will use use BOW and TF-IDF, with unigrams and bi-grams, to perform feature extraction and compare the resulting model performances. Both models were trained on the training set and the trained model was then used to transform both the training and test sets into numerical features. It is important to **only** train the feature extraction models on the training set to simulate real-world cases which are almost certainly going to include words the model has never seen before.

<u>Model Building</u>

For both BOW and TF-IDF generated features, 3 different classification algorithms were assessed: 1) Logistic Regression, 2) Random Forest Classifier, and 3) Multinomial Naive Bayes. All training accuracy values are cross-validated over 5 folds.

Bag-of-Words			
	Logistic Regression	**Random Forest**	**Naive Bayes**
Training Accuracy	0.86	0.81	0.75
Test Accuracy	0.86	0.82	0.75
Average Precision	0.86	0.82	0.75
Average Recall	0.86	0.82	0.75

TF-IDF			
	Logistic Regression	**Random Forest**	**Naive Bayes**
Training Accuracy	0.85	0.78	0.74
Test Accuracy	0.85	0.80	0.75
Average Precision	0.85	0.80	0.76
Average Recall	0.85	0.80	0.75

The logistic regression model performs best with the BOW model with an accuracy, average precision, and average recall of 86%.

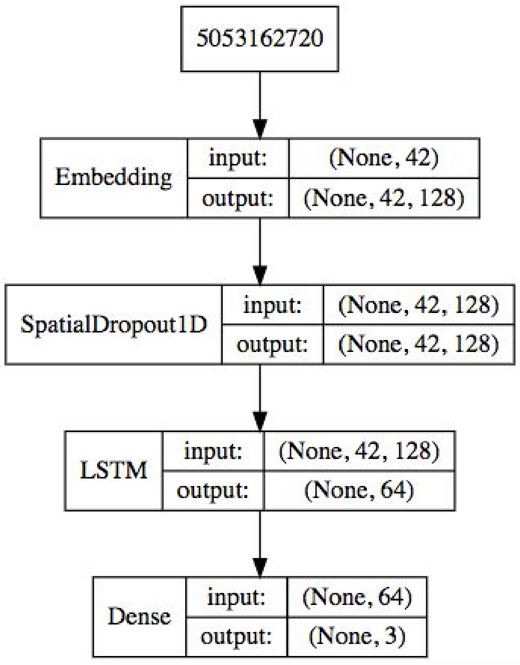
Recall that we removed stopwords as part of our pre-processing steps. While this is often considered standard practice, the removal of stopwords doesn't always lead to enhanced model performance. We can confirm whether the removal of stopwords is necessary for our data by re-running the logistic regression model WITH the stopwords and comparing the metrics with our previous results. Doing so resulted in a very slightly lower training accuracy of 0.85. We will therefore remove the stopwords from our final model.

Also recall that when generating our numerical features, we used 1 and 2-grams. This means that words like “native” and “american” are not just treated as separate features but “native american” will also be treated as 1 feature. We also tested adding tri-grams to the BOW model to capture additional relationships but doing so did not result in any improvement in model performance.

After hyperparameter tuning, the final logistic regression model was found to have an optimum ‘C’ parameter of 1.0 and metrics as listed in the table above.

Deep Learning

We also used Keras to develop a Long short-term memory (LSTM) model to see if a deep learning model would perform better than logistic regression. To prepare text data for our deep learning model, we transformed each tweet into a sequence by mapping every word in the tweet to an integer index. One requirement of Keras is that all sequences have to be of equal length. Since not all tweets are of the same length, we defined a maximum length to be the length of the longest tweet. All tweets which are smaller than this length were then padded with zeros. Target labels were re-encoded and the following LSTM model architecture was built.



The model was fit over 5 epochs resulting in a training accuracy of 95%, a validation accuracy of 87% and a test accuracy of 88%. The high training accuracy and lower validation and test accuracy show signs of overfitting. The average precision and recall scores were also 0.88.

While the LSTM model does result in a very slightly higher validation accuracy of 87%, logistic regression results in comparable accuracy (86%) with no signs of overfitting. It is also easier to understand and interpret than a deep learning model and is, therefore, our final choice for the sentiment analysis predictor.

Topic Modeling For Each Candidate

While predicting the sentiment of tweets is useful, this information can be better utilized if we can understand the topics being discussed in the tweets. Since there is no easy way to label the topics for a large number of tweets, unsupervised learning is currently the best way to find patterns in the data and cluster words of similar context together.

We can start by using a basic Bag-of-Words model to try and identify key topics based on term frequencies. In other words, we can look at the most common words used in the tweets about the candidates to identify topics, similar to the concept of word clouds. Unfortunately, there didn't seem to be an emergence of any clear topics simply by looking at the most common words.

The next step is to try a more complicated model. We will try 2 models: Non-negative Matrix Factorization (NMF) and Latent Dirichlet Allocation (LDA). NMF is an unsupervised learning algorithm that simultaneously performs dimensionality reduction and clustering. We can use it in conjunction with TF-IDF to model topics across the tweets. LDA is also an unsupervised learning algorithm but while NMF is a linear-algebraic model, LDA is based on probabilistic graphical modeling. We can use LDA with a simple Bag-of-Words model to extract topics.

Note that while Gensim is often used for topic modeling, we will opt to use Scikit-Learn because it enables us to perform both NMF and LDA. Based on the documentation, we use TF-IDF to perform feature extraction for NMF and Bag-of-Words (i.e. CountVectorizer in sklearn) for LDA. For both LDA and NMF, the number of topics and n-gram ranges are hyperparameters that need to be chosen by the user. We achieved the most coherent results with NMF, with 5 topics and an n-gram range of 1 to 3 words. Below are the topics and top 10 words output from both the NMF models for each candidate. Note that the output are simply words grouped together - it is up to us to manually assign labels to the topic based on those words.

NMF on Tweets about Bernie Sanders		
Topic	**Words**	**Manually Assigned Label**
0	'like', 'berniesanders', 'vote', 'president', 'people', 'democrat', 'one', 'candidate', 'support', 'want'	*Candidacy for president*
1	'contribute', 'contribute notmeus', 'notmeus', 'join', 'please', 'email', 'please donate', 'dedicate', 'donation', 'donate'	*Campaign for Donations*

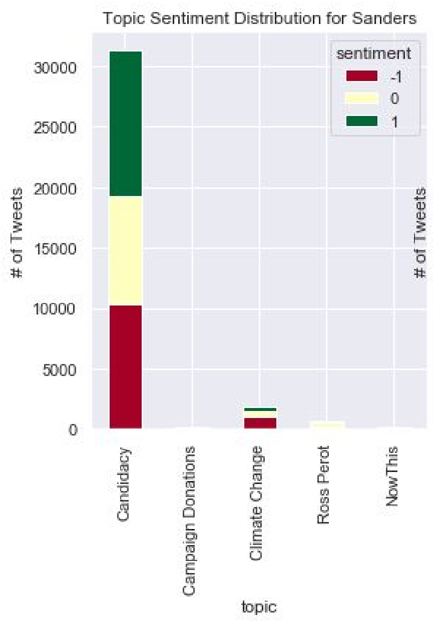
2	'climate', 'emergency', 'declare', 'aoc', 'declare climate', 'climate change', 'change', 'climate emergency', 'congress declare', 'congress'	*Climate Change*
3	'perot', 'ross', 'sword', 'ross perot', 'give', 'perot give', 'ross perot give', 'give sword', 'perot give sword', 'gift'	*Ross Perot's gift to Bernie Sanders (sword)*
4	'worry overpromising', 'overpromising', 'worry', 'nowthis', 'worry overpromising nowthis', 'overpromising nowthis', 'berniesanders worry', 'berniesanders', 'notmeus', 'feelthebern'	*NowThis's Interview of Bernie Sanders*

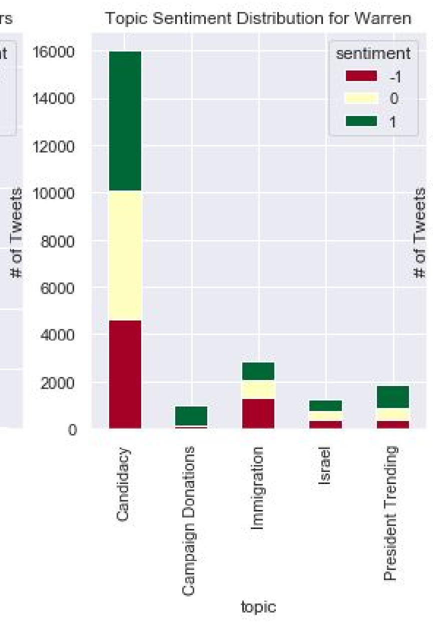
NMF on Tweets about Elizabeth Warren		
Topic	**Words**	**Manually Assigned Label**
0	'president', 'never', 'never president', 'president president', 'sound', 'like', 'presidentelizabethwarren', 'good', 'make', 'love'	*Candidacy for president*
1	'join support', 'join', 'support', 'donate', 'support president', 'candidate', 'liz', 'let', 'donate president', 'call'	*Campaign for Donations*
2	'cross border', 'cross', 'illegally', 'border', 'border illegally', 'cross border illegally', 'illegally law', 'border illegally law', 'law', 'criminal offense'	*Immigration/Border Policy*
3	'gaza', 'occupation', 'end', 'israel', 'end israels occupation', 'end israels', 'israels occupation', 'israels', 'occupation gaza', 'israels occupation gaza'	*Israel*
4	'trend', 'president trend', 'president', 'see', 'see president trend', 'see president', 'twitter', 'love', 'trend twitter', 'president trend twitter'	*"President Warren" trending on Twitter*

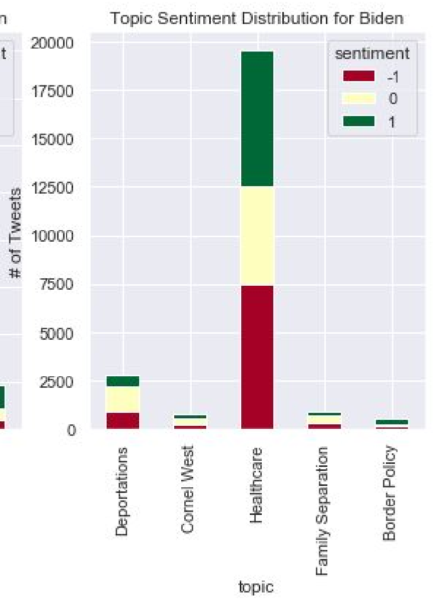
NMF on Tweets about Joe Biden		
Topic	**Words**	**Manually Assigned Label**
0	'refuse apologize', 'deportation number', 'refuse apologize high', 'high deportation number', 'high deportation', 'apologize high deportation', 'apologize high', 'number obama', 'deportation number obama', 'obama year'	*Refusing to Apologize for Deportations during Obama's Administration*
1	'cornel', 'cornel west', 'west', 'take cnn', 'cornel west take', 'west take', 'west take cnn', 'cnn', 'take', 'shot support'	*Cornel West*
2	'care', 'health', 'plan', 'health care', 'health care plan', 'care plan', 'president', 'like', 'bidens', 'act'	*Health Care Plan*
3	'family separation', 'separation', 'need family separation', 'need family', 'gaffe', 'family', 'immigration need', 'immigration need family', 'gaffe immigration need', 'gaffe immigration'	*Immigration/Family Separation*

4	'respect', 'border', 'respect border', 'respect border contain', 'border contain', 'border contain wall', 'contain wall', 'contain', 'wall', 'declare respect border'	*Border Policy*

We can also look at the sentiment distribution around each topic for each candidate. For example, we can see that there is more negative sentiment around Warren's immigration policy but more positive sentiment around tweets about campaign donations (makes sense!).


Topic Sentiment Distribution for Sanders
sentiment
# of Tweets
Candidacy
Campaign Donations
Climate Change
Ross Perot
NowThis
topic


Topic Sentiment Distribution for Warren
sentiment
# of Tweets
Candidacy
Campaign Donations
Immigration
Israel
President Trending
topic


Topic Sentiment Distribution for Biden
sentiment
# of Tweets
Deportations
Cornel West
Healthcare
Family Separation
Border Policy
topic