Twitter Sentiment Analysis of Electoral College in U.S.

~ Sneha Krishnamurthy

1. Introduction:

Twitter is a treasure trove of sentiments. The goal of this project is to predict Twitter users' political sentiment towards electoral college in U.S. for the given Twitter post using Python based Machine learning NLP Classification model. Many Americans are critical of the Electoral College, an attitude that seems to have intensified since Donald Trump defeated Hillary Clinton in the 2016 presidential election despite losing the popular vote. The data used for this model is obtained by using Tweepy to scrape Twitter API for data collection using the keyword "electoral college".

Sentiment Analysis is simply gauging the feelings behind a piece of content..It maps and measures the relationships and flows of information between people, communities and organizations In this work, we will build a machine learning classifier that can detect sentiment to classify tweets (pieces of text) as positive, negative or neutral by analyzing the feeling that Twitter users have towards "electoral college"

2. Problem:

This project explores on predicting how positive, neutral or negative each tweet is about electoral college. What is being said about Electoral College on social media? How has the sentiment changed over the years? We also want to understand the overall feelings towards Democrats and Republicans when it comes to the topic of electoral college.

3. Clients:

Sentiment Analysis models can be used by election campaign officials, election committees and also the general public to learn about the public sentiment towards electoral college. This study seeks to display helpful information about electoral college that even the candidates may consider in future while on the campaign trail.

4. Data:

This project uses Tweepy to access Twitter API for data collection using the keyword “electoral college”.

After dropping columns such as twitter handle and usernames, to comply with Twitter API policy, we were left with 47182 rows and 13 columns in the dataframe.

4.1 Data Dictionary:

The columns that are included are:

Attribute	Description
User_Id	The integer representation of the unique identifier for the Tweet.
Total Tweets	total number of Tweets the account has posted
Favourites_Count	Indicates approximately how many times this Tweet has been liked by Twitter users.
Followers	number of followers
User_Verified	Boolean value True if the blue verified badge is present on Twitter which lets people know that an account of public interest is authentic.
User Location	Place associated with tweet
Date of Tweet	UTC time when the Tweet was created.
Tweet Id	The numerical ID of the desired Tweet.
Tweet Text	The actual UTF-8 text of the status update
Language	When present, indicates a BCP 47 language identifier corresponding to the machine-detected language of the Tweet text
Tweet Source	Utility used to post the Tweet
Tweet Retweet	Number of times this Tweet has been retweeted

Tweet Reply To Id	If the represented Tweet is a reply, this field will contain the integer representation of the original Tweet’s author ID.

After basic cleaning of data extracted from the Twitter api, sentiment score was generated for each tweet, which would be our target variable. To create target variable, we have compared several sentiment analyzer tools which are widely available for classifying the data such as VADER, Textblob, SentiWordNet lexicon from NTLK, StanfordCoreNLP, Afinn as provided in this link

StanfordCoreNLP was used to label our tweets dataset since it is designed to help evaluate a model’s ability to understand representations of sentence structure, rather than just looking at individual words in isolation.

5. Exploratory Data Analysis:

In this project, we’ll be retaining only English language tweets. After filtering the data-frame with only english language tweets, we were left with 46831 rows of data. The first step in building the model is cleaning the data obtained and analyzing the data by applying EDA techniques using Python and other libraries such as the natural language toolkit (nltk), gensim, scikit-learn and spaCy.

5.1 Character length:

The average character count was approximately 128, which is well within the limit of overall average tweet length of around 140 characters. As far as Twitter is concerned, every single character in a Tweet counts as one for the purposes of the character count which includes letters, numbers, spaces,letters with accent marks, mentions,hashtags and other symbols.There is only one exception to this rule which is that Twitter uses its own URL shortening service, any website address you post in a tweet will count as 22 characters, regardless of whether it originally was longer or shorter than that. Even extremely basic features such as character counts can prove to be very useful to find outliers. Looks like we do not need to worry about that for now since the character length is well within the limit.

Most of the tweets in our dataset have some elements that we do not want in our character counts. Below is a conversations at glance extracted from “Tweet Text” column:

```
"'We\\xe2\\x80\\x99re still skewed by population but not quite near as much. TGI Electoral College...
\\n#ElectionResults2019\\xe2\\x80\\xa6 #ElectionResults2019 #ElectoralCollege #Election2020
#YangGang #FeelTheBern #TrumpTrain https://t.co/5ilhukuqnr'"
```

As we can see, we have to remove 'b' prefix which is a decoded string in Python to stop it from interfering in future coding. The hexadecimal representation \xe2\x80\x99 of the Unicode character U+2019 is actually the right single quotation mark. We also see strange patterns of characters "\xe2\x80\xa6" which are UTF-8 BOM (Byte Order Mark)."The UTF-8 BOM is a sequence of bytes that allows the reader to identify a file as being encoded in UTF-8."

It looks like HTML encoding has not been converted to text as can be seen at row #2, and ended up in text field as '&amp','&quot',etc. Decoding HTML to general text was my next step of data preparation. I used BeautifulSoup for this.

After basic text cleaning, the tweet conversations look as below:

"'We're still skewed by population but not quite near as much. TGI Electoral College... #ElectionResults2019\\xe2\\x80\\xa6 #ElectionResults2019 #ElectoralCollege #Election2020 #YangGang #FeelTheBern #TrumpTrain https://t.co/5ilhukuqnr'"

URL links, @ symbols and hashtags need to be included in character count and was preprocessed later.

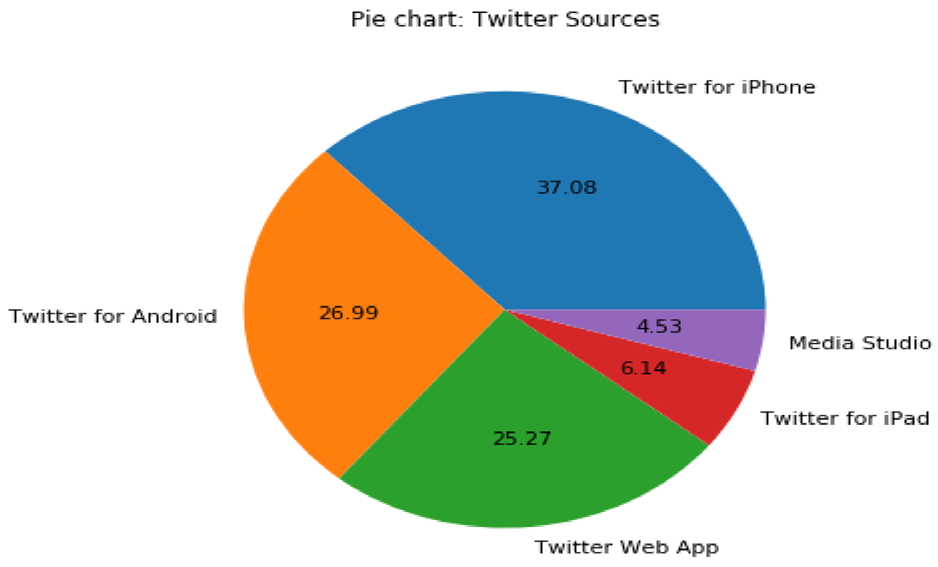
5.2 Emoticons:

We also found out that there were no emojis left when we ran a function to check if any emojis are left in text column.

5.3 Twitter Sources:

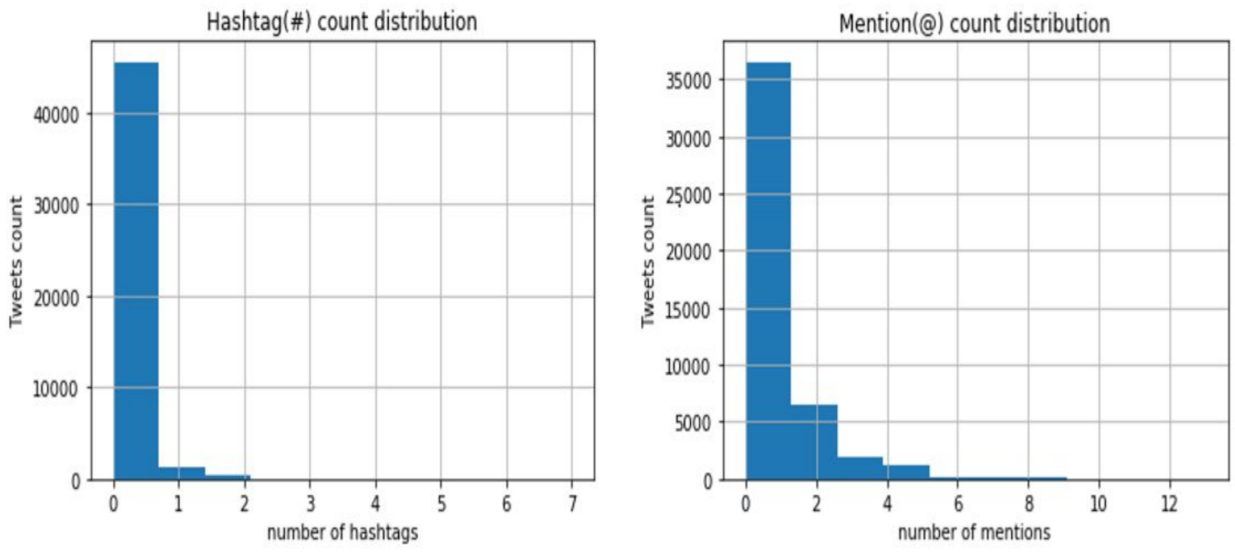
When we analyzed twitter sources, we realized that basically this twitter dataset has four major sources such as “Twitter for iPhone”, “Twitter for Android”, “Twitter Web App” ,”Twitter for ipad” and the rest such as “AshburnTimes”, “Hootsuite Inc.”, etc., were consolidated as ” Media

Studio" as shown in the pie chart below:

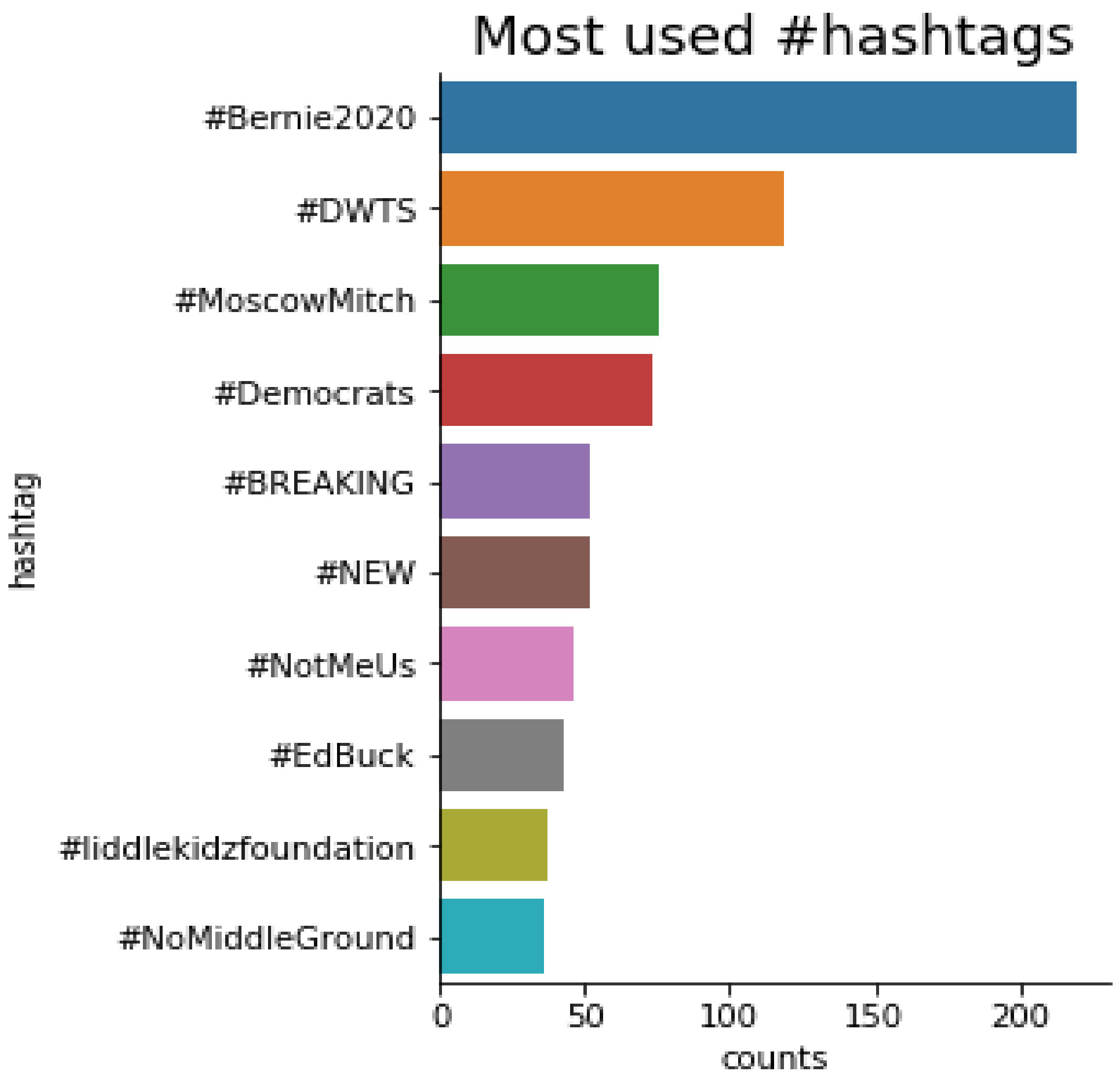


5.4 Hashtags and Mentions:

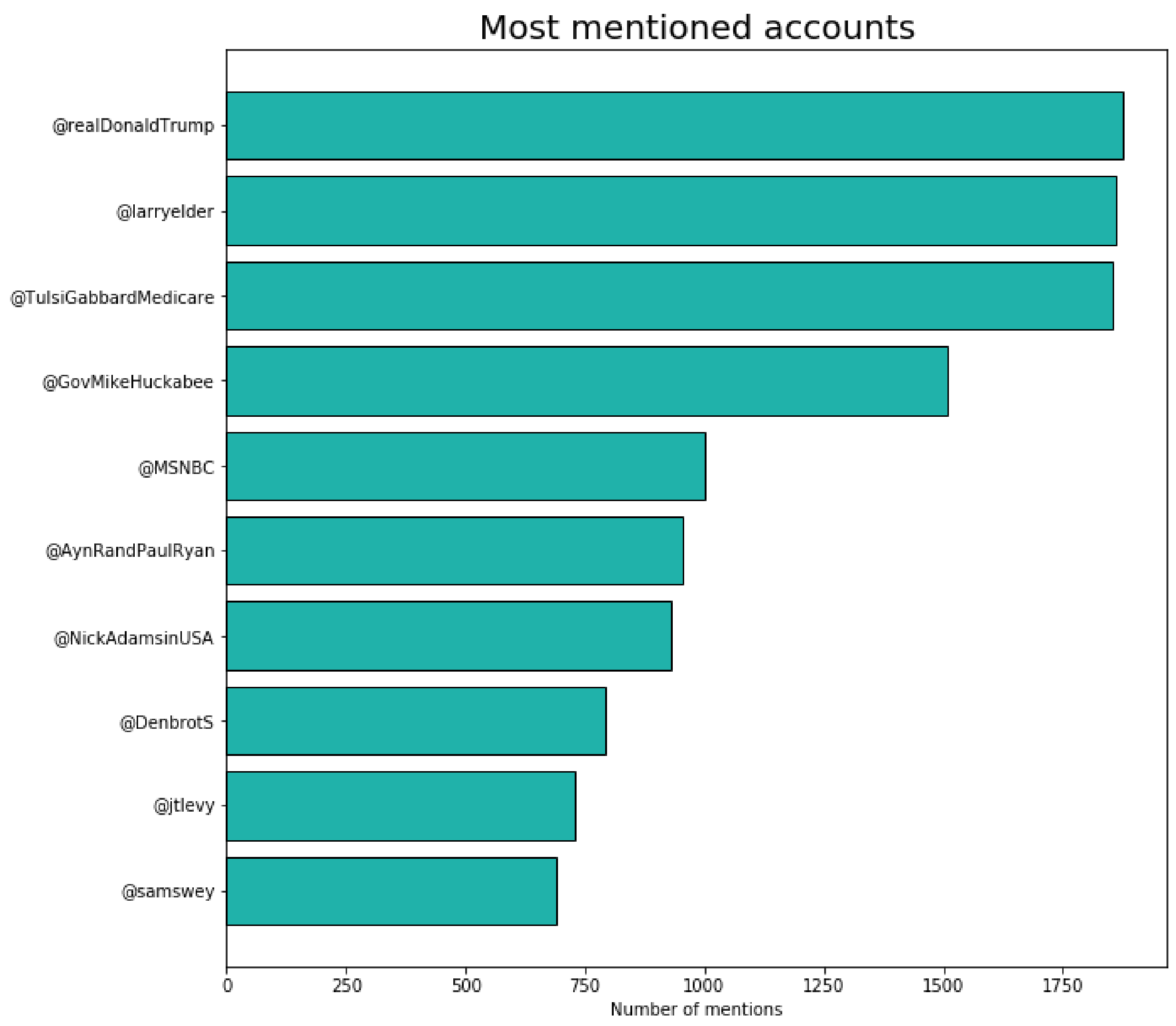
Comparison of the distribution of hashtags and mentions in tweets from this dataset:



It looks like the number of mentions is more than the number of hashtags in our dataset.Below is a graph created using Seaborn to visualize most used hashtags:



From this figure it can be clearly seen that #Bernie2020 is the most used hashtag. #NotMeUs also is also a political organization looking to further the progressive cause by following Bernie's lead. We can guess that many of Bernie's supporters are actively opposing electoral college.



It is no surprise that @realDonaldTrump leads the list.

6. Data Preprocessing:

It is important to check if we have **duplicates** which is common in tweets because of the RT tag (Retweets), After **dropping duplicates**, we were left with 18367 rows of data.

Regular expressions were used to get rid of

1) special characters, numbers, punctuation, etc,.
2) replace username-tags with blank space
3) replace hashtags with blank space
4) Retain words with only 3 characters are more
5) Retain alphabetic words

Using a function, the text was further cleaned and the following tasks were performed.

1) Reduce all letters to **lower- case**
2) Stop words are removed using ntlk library and additional stop words
3) Tokenize text using word_tokenize
4) Stem the words using ntlk Porterstemmer
5) Lemmatize all tokens using WordNetLemmatizer.
6) Detokenize lemmatized tokens for vectorizing

Below is an example to show the difference in the raw text and processed tweet:

Tweet Text #21 as is:
'@RepKevinBrady @GOPLeader @KPRC2Khambrel We\xe2\x80\x99ll find out with public hearings. Let\xe2\x80\x99s be clear. Trump lost by 3 milli\xe2\x80\xa6 https://t.co/VXdYuJdjQK'

Text cleaned and filtered with stop words:
find public hearings let clear trump lost milli

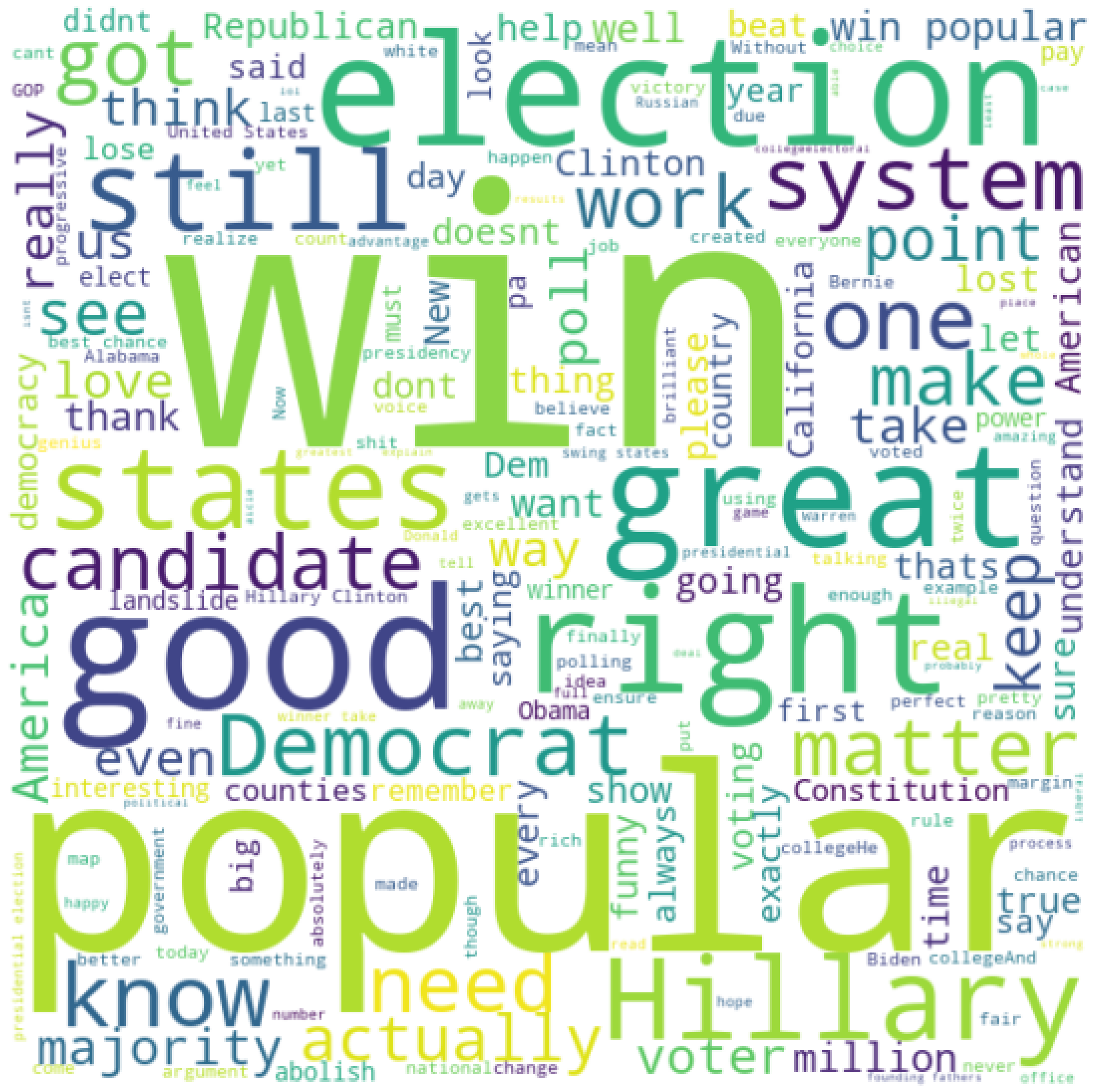
processed text - stemmed, Lemmatized and de-tokenized:
'find public hear let clear trump lost milli'

7. Data Visualization:

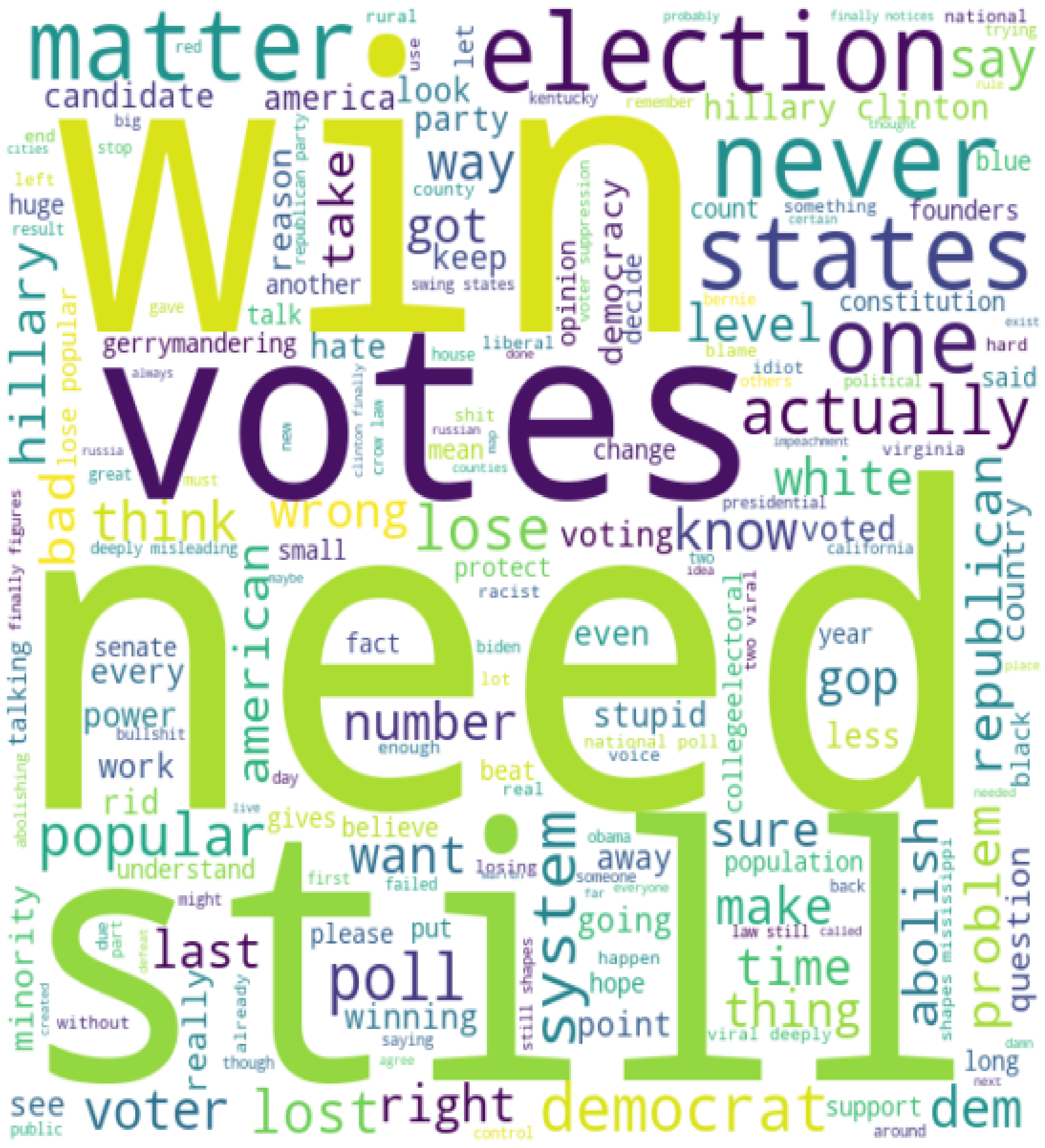
A word cloud represents word usage in a document by resizing individual words proportionally to its frequency and then presenting them in a random arrangement. The textual analysis of our tweets provides a general idea of what kind of words are frequent in our corpus.

The word cloud visualization of the tweets having positive sentiment, neutral sentiment and negative sentiment is provided below:

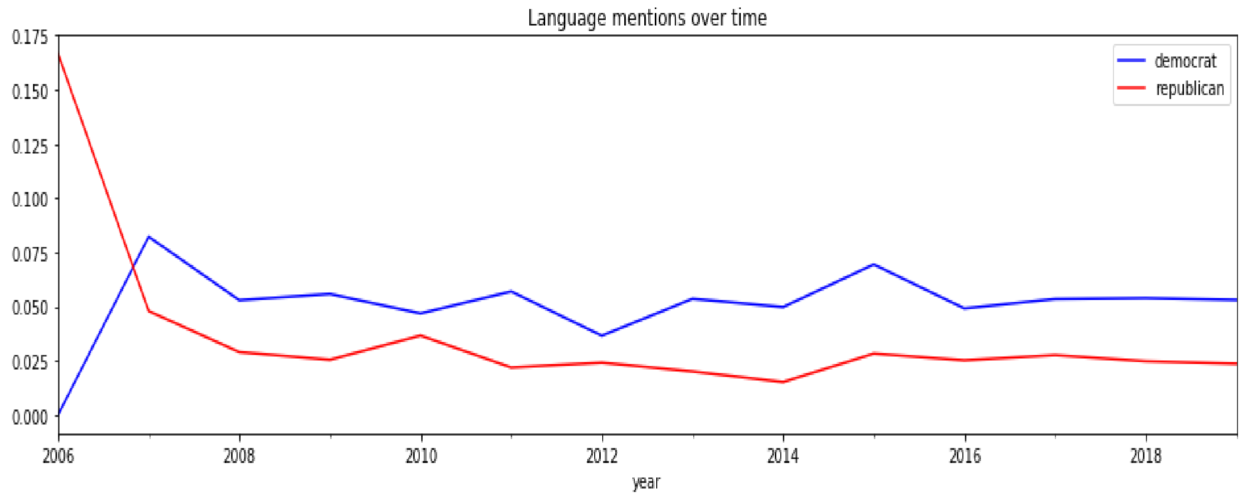
Word cloud of positive text



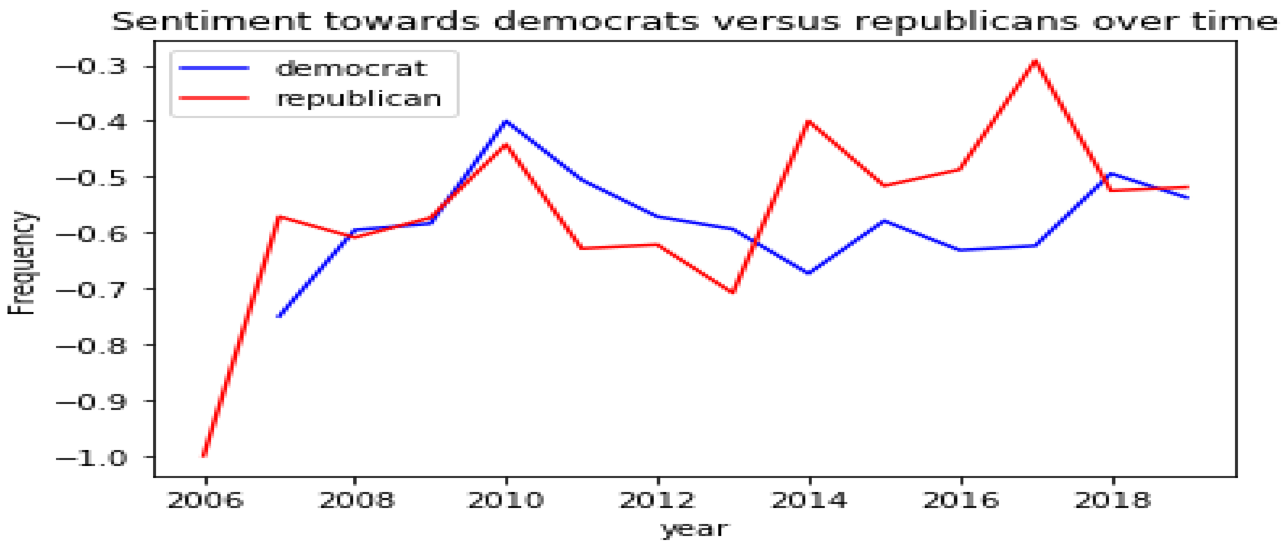
Word cloud of negative text



Twitter has a reputation as the most political of all the social media platforms. According to Pew survey, 36% of U.S.-based Twitter users are Democrats and only 21% of Twitter users are Republicans. Here, we are interested to capture the number of times the language mentions of “democrat” and “republicans” occured in our dataset containing tweets associated with electoral college and compare it to sentiment (opinion trend) associated with language mentions in time series.

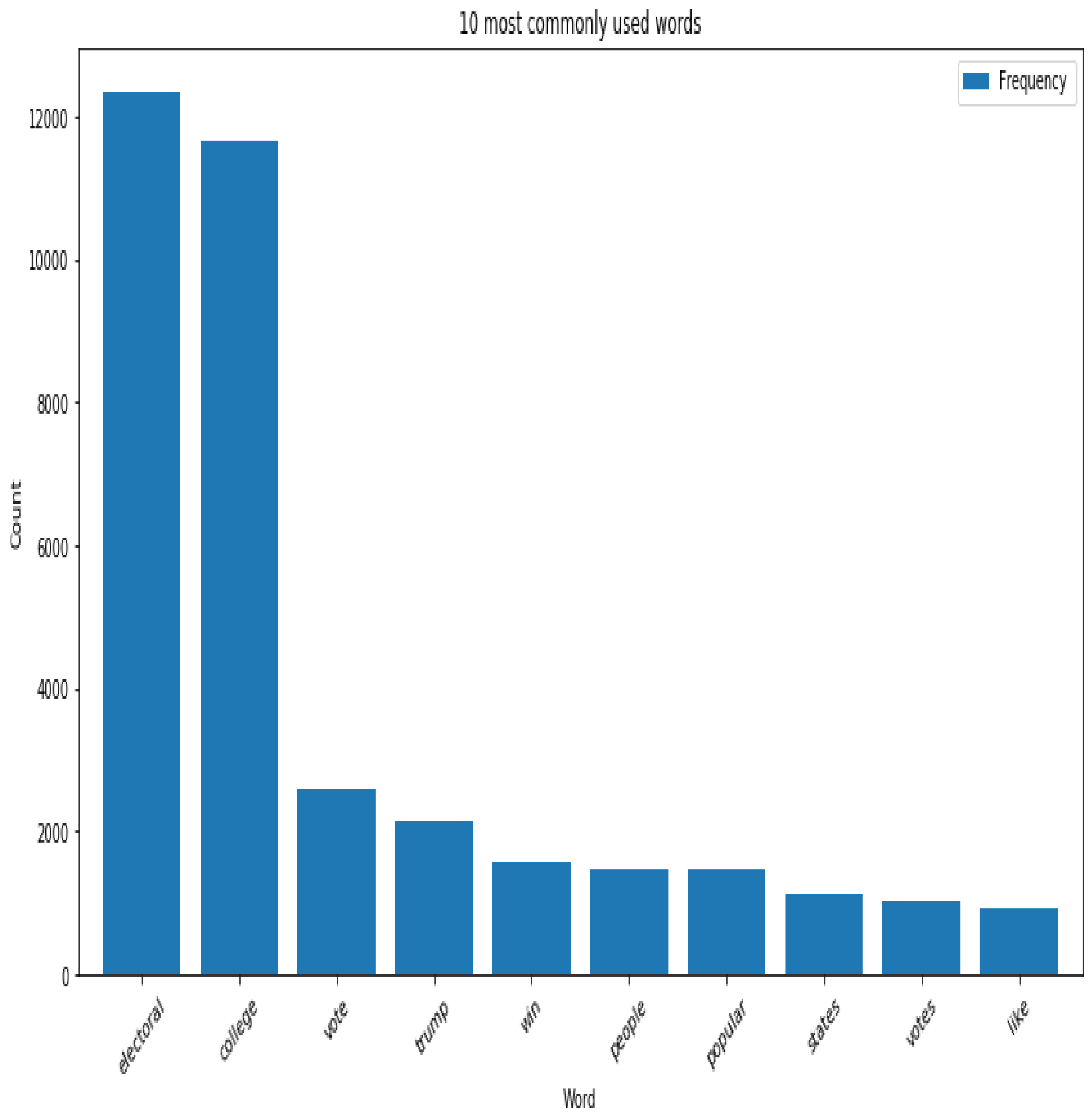


As we can see in the plot above, the term “republican” has been mentioned mostly fewer number of times compared to the term “democrat” except in years 2006 and 2007.

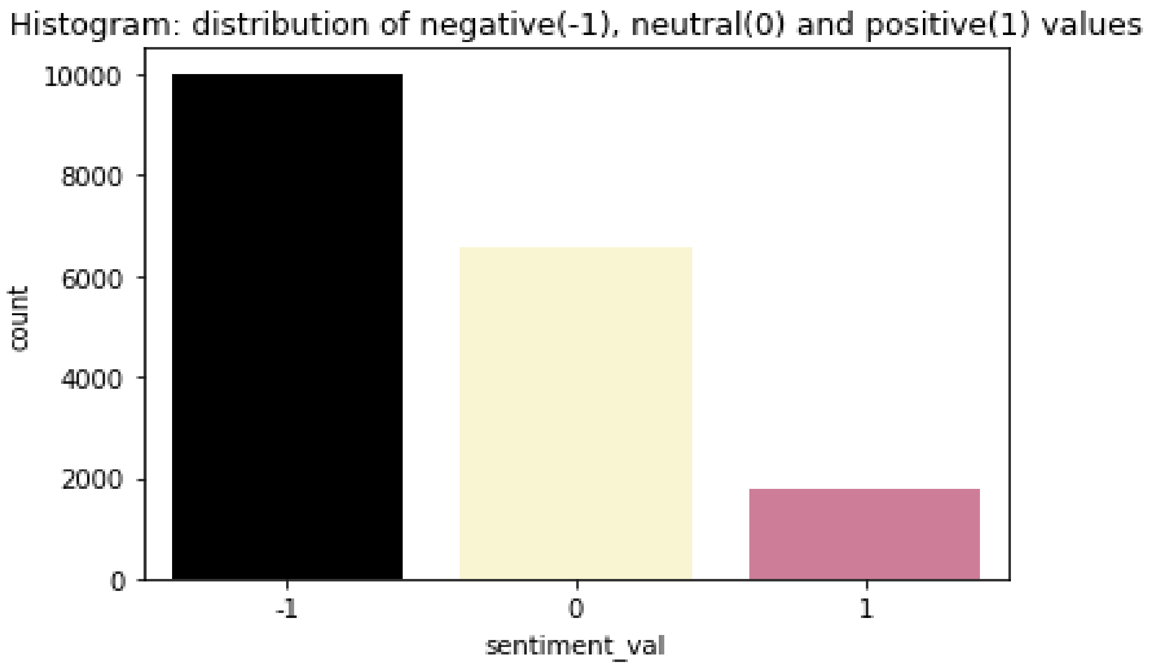


When we compare the opinion trend as seen in the plot above, surprisingly, sentiment toward republicans looks slightly better than democrats just after 2013.

Now, let's explore the 10 most popular words in our data-frame:

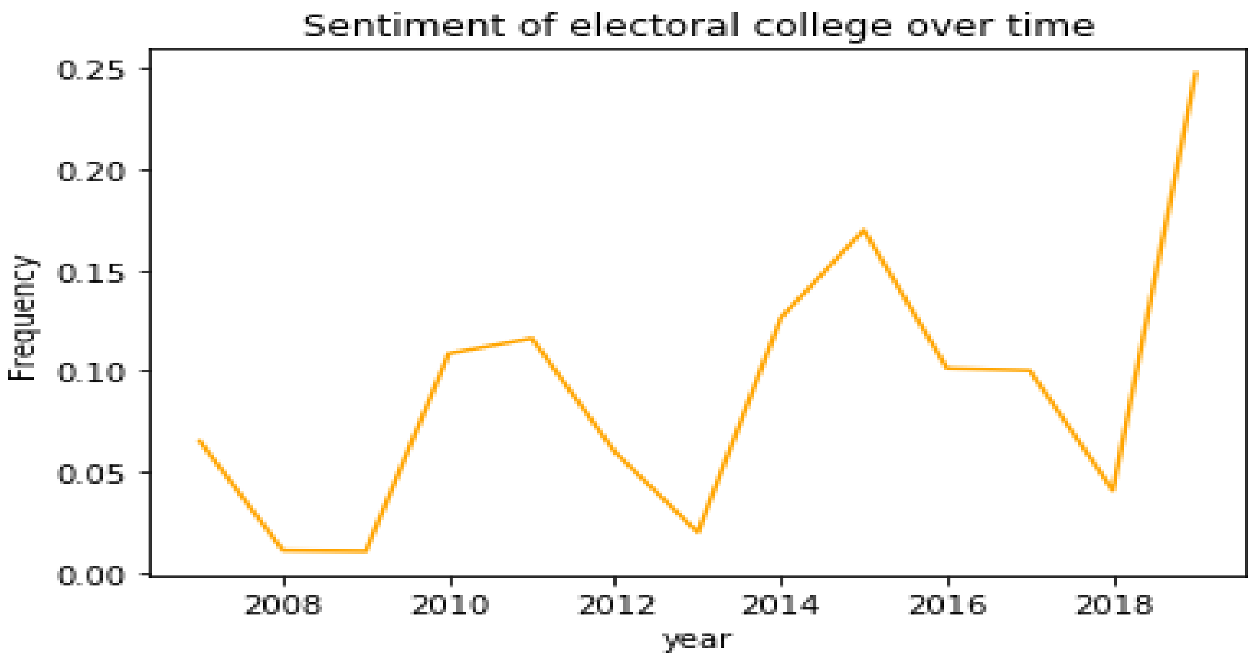


Label distribution of sentiments can be seen in the plot below:

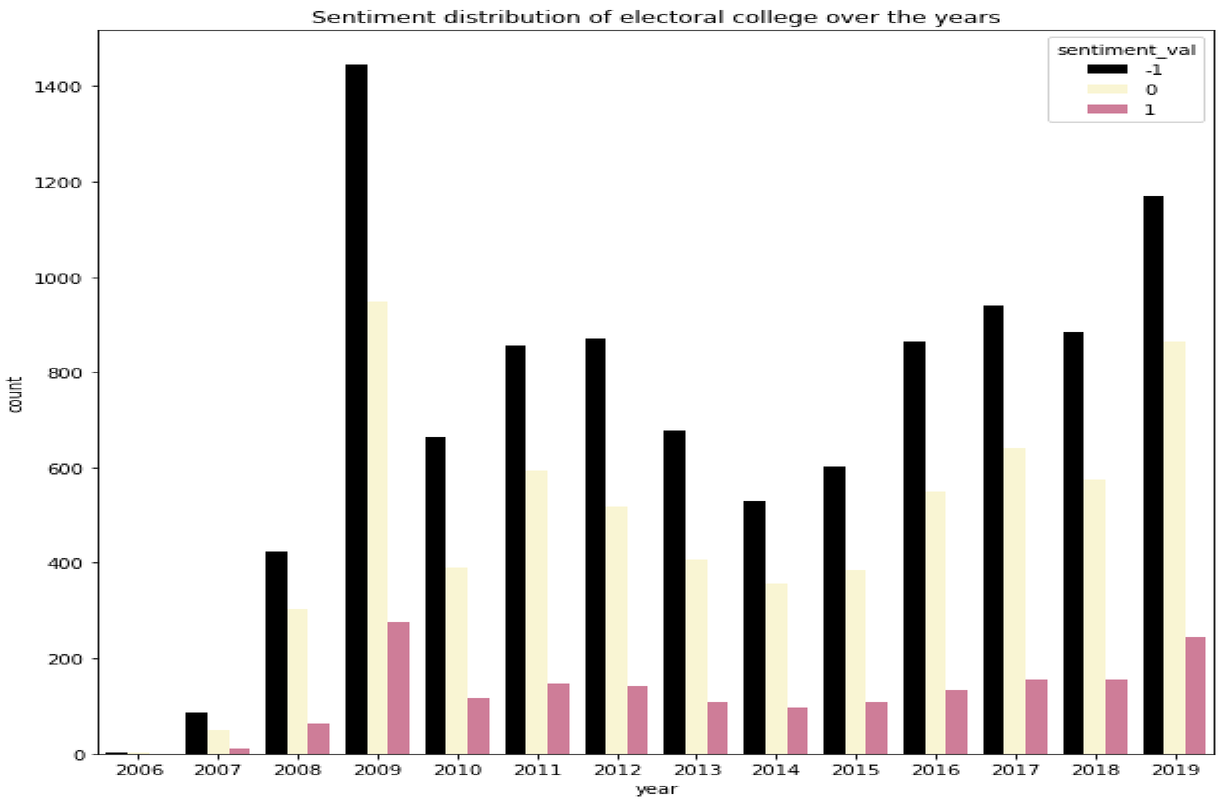


We have more tweets that are labelled negative compared to neutral and negative tweets.

Next, let’s analyse how sentiment about electoral college change over time in the plot below:



We can see from the plot above that the average sentiment towards electoral college has been negative for the large part comparitively, it peaked in 2015 suggesting it could have been due to 2016 election results. We will now segregate tweets based on sentiment counts and compare it over the years.



8. Inferential Statistics:

Based on the plot above, we can see that there was a noticeable variations in polarity of average sentiment throughout the years. I would like to to test whether this hypothesis is commensurate with the data, especially from 2014 to 2016. The dataset has twitter data from from 2006 to 2018. The samples derived based on years are independent of each other and are random. I would be performing a two sample Bootstrap hypothesis test with null hypothesis being that there is no difference in means with regards to the average sentiment polarity.

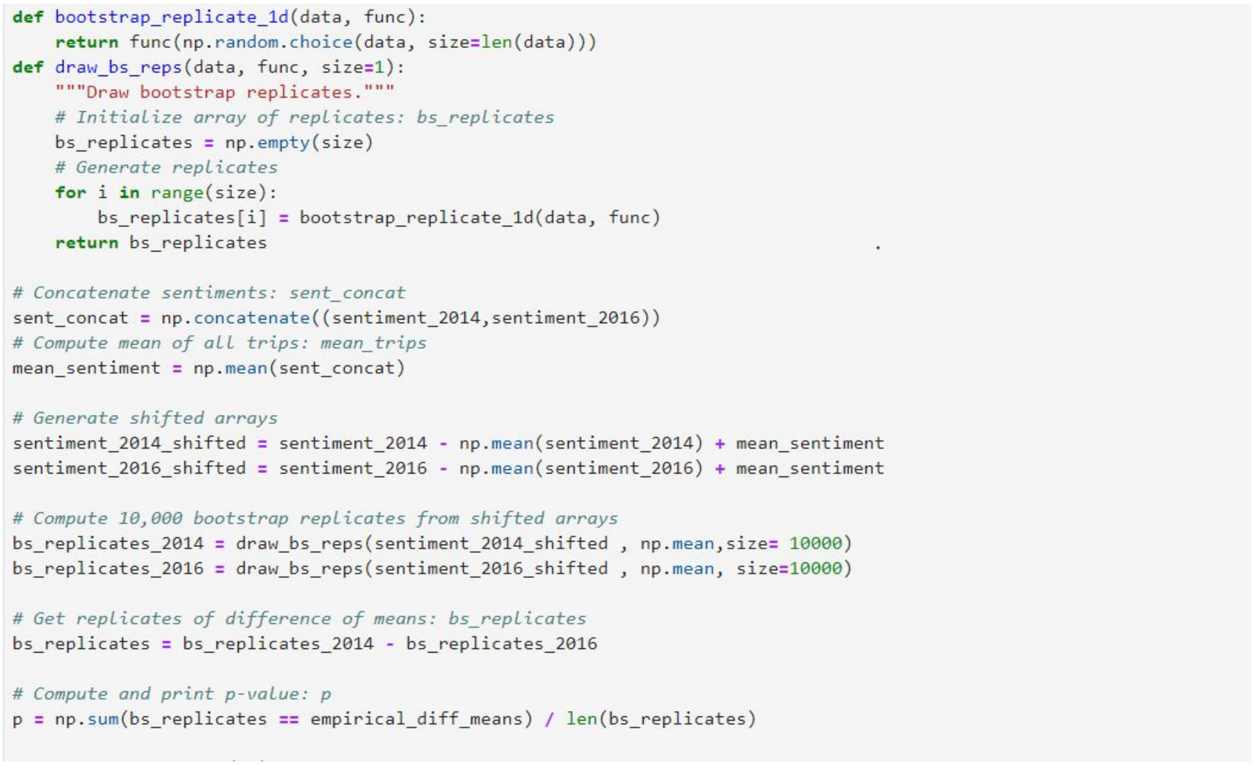
Null Hypothesis:

$Ho \Rightarrow \ \mu 1 = \mu 2$

Alternate Hypothesis:

$Ha \Rightarrow \mu 1 \neq \mu 2$

The average sentiment polarity for the year 2014 was -0.44, and that of year 2016 was 0.47 with an empirical difference of 0.042. It is possible this observed difference in mean was by chance. We will compute the probability of getting at least a 0.042 difference in average sentiment polarity under the hypothesis that the average sentiment polarity in both years are identical. We want to test the hypothesis that the year 2014 and 2016 have the same average sentiment polarity with respect to electoral college using the two-sample bootstrap test. Here, we shift both arrays to have the same mean, since we are simulating the hypothesis that their means are, in fact, equal. We then draw bootstrap samples out of the shifted arrays and compute the difference in means. This constitutes a bootstrap replicate, and we generate many of them. The p-value will be the fraction of replicates with a difference in means greater than or equal to what was observed. Two-sample bootstrap hypothesis test for difference of means was performed using the following code:



```python
def bootstrap_replicate_1d(data, func):
    return func(np.random.choice(data, size=len(data)))
def draw_bs_reps(data, func, size=1):
    """Draw bootstrap replicates."""
    # Initialize array of replicates: bs_replicates
    bs_replicates = np.empty(size)
    # Generate replicates
    for i in range(size):
        bs_replicates[i] = bootstrap_replicate_1d(data, func)
    return bs_replicates

# Concatenate sentiments: sent_concat
sent_concat = np.concatenate((sentiment_2014,sentiment_2016))
# Compute mean of all trips: mean_trips
mean_sentiment = np.mean(sent_concat)

# Generate shifted arrays
sentiment_2014_shifted = sentiment_2014 - np.mean(sentiment_2014) + mean_sentiment
sentiment_2016_shifted = sentiment_2016 - np.mean(sentiment_2016) + mean_sentiment

# Compute 10,000 bootstrap replicates from shifted arrays
bs_replicates_2014 = draw_bs_reps(sentiment_2014_shifted , np.mean,size= 10000)
bs_replicates_2016 = draw_bs_reps(sentiment_2016_shifted , np.mean, size=10000)

# Get replicates of difference of means: bs_replicates
bs_replicates = bs_replicates_2014 - bs_replicates_2016

# Compute and print p-value: p
p = np.sum(bs_replicates == empirical_diff_means) / len(bs_replicates)
```

.We got a p-value of 0, which suggests that there is a statistically significant difference in the average sentiment polarity from 2014 to 2016. But it is very important to know how different they are! We got a difference of 0.042 between the means. The difference doesn't seem to be that substantial and there might be few situations where sentiment polarity varies.

Since p-value is < 0.05, the difference is statistically significant rendering evidence to reject the hypothesis with 95% confidence.

Future work:

Feature selection

train/test split

Modeling: Multi Label Text Classification

Deliverables:

Powerpoint presentation

Python code used for data preprocessing, analysing, modeling in Github link.

A PDF report on the findings.