## Reddit Sentiment Analysis

Demonstration on how to create a word cloud and sentiment analysis using the reddit API

```
In [ ]: #pip install praw
  In [2]: #pip install wordcloud
In [18]: import praw
                           credentials
               import pandas as pd
from wordcloud import WordCloud
import matplotlib.pyplot as plt
               Create a Reddit account, then go to https://www.reddit.com/prefs/apps
 This subreddit is here: https://www.reddit.com/r/socialjustice101/top/?t=all
In [86]: subreddit = reddit.subreddit('climatechange').top('all', limit=1000)
               C:\Users\matth\AppData\Local\Temp\ipykernel_1272\2092646888.py:1: DeprecationWarning: Positional arguments for 'BaseListingMixin.top' will no longer be supported in PRAW 8. Call this function with 'time_filter' as a keyword argument.

subreddit = reddit.subreddit('climatechange').top('all', limit=1000)
co_subreddit = reddit.subreddit('climatechange').hot(limit=None)
               for post in list(co_subreddit):
    reddit_data['title'].append(post.title)
    reddit_data['link'].append(post.permalink)
    if post.author is None:
        reddit_data['author'].append('')
                           reddit_data['author'].append(post.author.name)
                     reddit_data['n_comments'].append(post.num_comments)
reddit_data['score'].append(post.score)
reddit_data['text'].append(post.selftext)
In [62]: import pandas as pd
In [63]: climate df = pd.DataFrame(reddit data)
In [64]: import nltk
               fd = nltk.FreqDist(' '.join(climate_df['title']).split())
               fd.most_common(20)
Out[64]: [('the', 217),
                [('the', 217),
('of', 172),
('of', 172),
('to', 170),
('climate', 135),
('in', 132),
('an', 114),
('a', 105),
('climate', 72),
('for', 72),
('for', 72),
('is', 69),
('on', 53),
('The', 48),
('How', 39),
('are', 39),
('that', 34),
('by', 32),
('for', 30),
('by', 32),
('form', 30),
('What', 28)]
 In [65]: nltk.download('stopwords')
               [nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\matth\AppData\Roaming\nltk_data...
               C:\Users\matth\AppData\Roaming\nltk_da [nltk_data] Package stopwords is already up-to-date! True
Out[65]:
In [66]: from nltk.corpus import stopwords
               stops = stopwords.words('english')
In [67]: words = ' '.join(climate_df['title']).lower().split()
cleaned_words = [w for w in words if w not in set(stops)]
fd = nltk.FreqDist(' '.join(cleaned_words).split())
In [70]: cleaned_fd = nltk.FreqDist(cleaned_words)
    cleaned_fd.most_common(10)
```

Word Cloud

1 of 2 4/1/2024, 1:22 PM

-1.00

-0.75 -0.50

-0.25

0.00

0.25

0.50

0.75

```
In [90]: subreddit = reddit.subreddit('climatechange').top('all', limit=100)
            C:\Users\matth\AppData\Local\Temp\ipykernel_1272\2520004515.py:1: DeprecationWarning: Positional arguments for 'BaseListingMixin.top' will no longer be supported in PRAW 8. Call this function with 'time_filter' as a keyword argument.

subreddit = reddit.subreddit('climatechange').top('all', limit=100)
for post in subreddit:
                 comments_dict['title'].append(post.title)
comments_dict['body'].append(post.selftext)
In [93]: wc = WordCloud().generate(' '.join(comments_dict['title']))
plt.imshow(wc, interpolation="bilinear")
plt.axis('off')
Out[93]: (-0.5, 399.5, 199.5, -0.5)
                                           global
                                                                     carbon
                                                   million heat
               climate crisis tree and world plant year year and sayplanting S
                                                                         world
                      warn pledge day climate
            Sentiment Analysis
 for post in subreddit:
    comments_dict['title'].append(post.title)
    comments_dict['body'].append(post.selftext)
In [44]: sentiment_df = pd.read_csv('AFINN-en-165.txt', sep='\t', names=['word', 'score'], index_col='word')
sentiment_dict = sentiment_df.to_dict()['score']
In [47]: import numpy as np
            title_sentiments = []
for title in climate_df['title']:
   words = title.lower().split()
   this_titles_sentiments = []
                     w in words:
if w in sentiment_dict.keys():
   this_titles_sentiments.append(sentiment_dict[w])
                     else:
this_titles_sentiments.append(0)
                 \verb|title_sentiments.append(np.mean(this_titles_sentiments))|\\
In [49]: climate_df['keyword_sentiment'] = title_sentiments
In [60]: climate_df['keyword_sentiment'].plot.hist(bins=30, title = 'Reddit Sentiment Anlysis')
Out[60]: <Axes: title={'center': 'Reddit Sentiment Anlysis'}, ylabel='Frequency'>
                                               Reddit Sentiment Anlysis
                300
                250
                200
            Frequency
150
                100
                  50
```

In [ ]:

2 of 2