Notebook9-checkpoint

May 22, 2020

[ ]:
```
# Text Analysis | SNA | Sentiment analysis
```

[89]:
```
#modules import

import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import itertools
import collections

import tweepy as tw
import nltk
from nltk import bigrams
from nltk.corpus import stopwords
import re
import networkx as nx

import warnings
warnings.filterwarnings("ignore")

sns.set(font_scale=1.5)
sns.set_style("whitegrid")
```

[90]:
```
#defining my keys

consumer_key= 'wDfiB8LIdPZkqngnZR2g9doVM'
consumer_secret= '9vGVZwosoZuNiSHpP7wY6WOFgpdNH21wF0eDcICOZRc10IIjCp'
access_token= '2735159228-UV7M2Wv9wZ5WEPx6lt9p4kdEnQ58euVXmKOhdZW'
access_token_secret= 'GHLlOmT8eOIFtQ3u664zf5aFX1rFdgontow9I6WZCywZf'
```

[91]:
```
auth = tw.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tw.API(auth, wait_on_rate_limit=True)
```

[92]:
```
# Create a custom search term cancer and gdp and hdi and define the number of␣
↪tweets
search_term = "#cancer+awareness -filter:retweets"

tweets = tw.Cursor(api.search,
                   q=search_term,
                   lang="en",
                   since='2018-11-01').items(1000)
```

[93]:
```
# grab and clean up 1000 recent tweets
```

[94]:
```
def remove_url(txt):
    """Replace URLs found in a text string with nothing
    (i.e. it will remove the URL from the string).

    Parameters
    ----------
    txt : string
        A text string that you want to parse and remove urls.

    Returns
    -------
    The same txt string with url's removed.
    """

    url_pattern = re.compile(r'https?://\S+|www\.\S+')
    no_url = url_pattern.sub(r'', txt)

    return no_url
```

[95]:
```
# Remove URLs
tweets_no_urls = [remove_url(tweet.text) for tweet in tweets]

# Create a sublist of lower case words for each tweet
words_in_tweet = [tweet.lower().split() for tweet in tweets_no_urls]

# Download stopwords
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))

# Remove stop words from each tweet list of words
tweets_nsw = [[word for word in tweet_words if not word in stop_words]
              for tweet_words in words_in_tweet]

# Remove collection words
collection_words = ['climatechange', 'climate', 'change']
```

```
tweets_nsw_nc = [[w for w in word if not w in collection_words]
                 for word in tweets_nsw]
```

```
[nltk_data] Downloading package stopwords to /home/haziz/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

[96]:
```
#Explore Co-occurring Words
```

[97]:
```
# Create list of lists containing bigrams in tweets
terms_bigram = [list(bigrams(tweet)) for tweet in tweets_nsw_nc]
```

[98]:
```
# View bigrams for the first tweet
terms_bigram[0]
```

[98]:
```
[('support', 'emerging'),
 ('emerging', 'musicians,'),
 ('musicians,', 'bring'),
 ('bring', 'power'),
 ('power', 'passion'),
 ('passion', 'people'),
 ('people', '#music'),
 ('#music', 'together'),
 ('together', 'raise'),
 ('raise', 'funds…')]
```

[99]:
```
# Original tweet without URLs
tweets_no_urls[0]
```

[99]:
```
'With the support of emerging musicians, we bring the power and passion of
people and #music together to raise funds… '
```

[100]:
```
# Clean tweet
tweets_nsw_nc[0]
```

[100]:
```
['support',
 'emerging',
 'musicians,',
 'bring',
 'power',
 'passion',
 'people',
 '#music',
 'together',
 'raise',
 'funds…']
```

[101]:
```
# Flatten list of bigrams in clean tweets
bigrams = list(itertools.chain(*terms_bigram))

# Create counter of words in clean bigrams
bigram_counts = collections.Counter(bigrams)
```

[102]:
```
bigram_counts.most_common(25)
```

[102]:
```
[(('raise', 'awareness'), 14),
 (('cancer', 'awareness'), 11),
 (('awareness', 'funds'), 10),
 (('sound', 'affects'), 7),
 (('awareness', 'month.'), 6),
 (('raising', 'awareness'), 5),
 (('skin', 'cancer'), 5),
 (('awareness', 'month'), 5),
 (('power', 'passion'), 4),
 (('skin', '#cancer'), 4),
 (('esophageal', 'cancer'), 4),
 (('#skin', '#cancer'), 4),
 (('.', '.'), 4),
 (('affects', 'raising'), 4),
 (('funds', 'technologies'), 4),
 (('technologies', 'potentially'), 4),
 (('potentially', 'prevent,'), 4),
 (('prevent,', 'diagnose,…'), 4),
 (('music', 'medium'), 4),
 (('medium', 'generating'), 4),
 (('generating', 'movement'), 4),
 (('movement', 'war'), 4),
 (('war', 'cancer.'), 4),
 (('cancer.', 'partner'), 4),
 (('partner', 'independent'), 4)]
```

[103]:
```
#data frames

bigram_df = pd.DataFrame(bigram_counts.most_common(25),
                             columns=['bigram', 'count'])

bigram_df
```

[103]:

	bigram	count
0	(raise, awareness)	14
1	(cancer, awareness)	11
2	(awareness, funds)	10
3	(sound, affects)	7
4	(awareness, month.)	6

```
5          (raising, awareness)      5
6                (skin, cancer)      5
7            (awareness, month)      5
8              (power, passion)      4
9                (skin, #cancer)     4
10         (esophageal, cancer)      4
11             (#skin, #cancer)      4
12                       (., .)      4
13           (affects, raising)      4
14        (funds, technologies)      4
15  (technologies, potentially)      4
16      (potentially, prevent,)      4
17       (prevent,, diagnose,…)      4
18               (music, medium)     4
19         (medium, generating)      4
20       (generating, movement)      4
21               (movement, war)     4
22                (war, cancer.)     4
23           (cancer., partner)      4
24       (partner, independent)      4
```

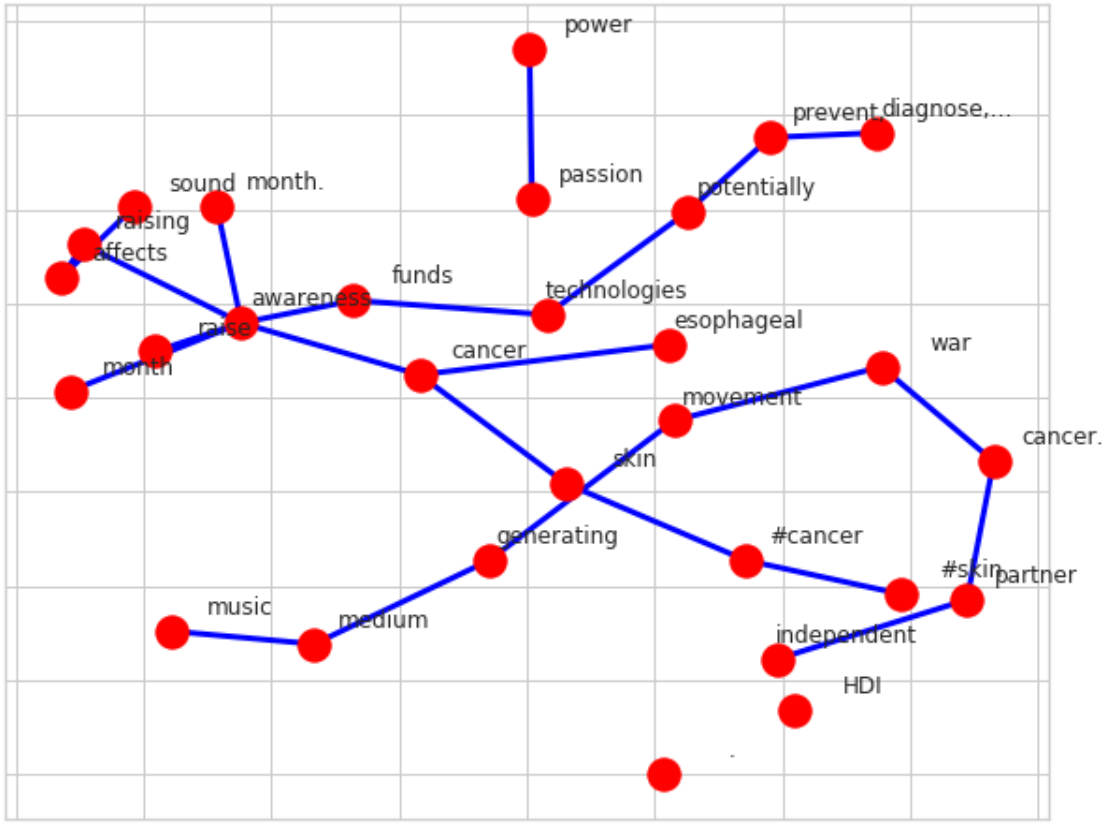
[104]:
```
#Visualize Networks of Bigrams
# Create dictionary of bigrams and their counts
d = bigram_df.set_index('bigram').T.to_dict('records')
```

[105]:
```
# Create network plot
G = nx.Graph()
```

[106]:
```
# Create connections between nodes
for k, v in d[0].items():
    G.add_edge(k[0], k[1], weight=(v * 10))

G.add_node("HDI", weight=100)
```

[107]:
```
fig, ax = plt.subplots(figsize=(10, 8))

pos = nx.spring_layout(G, k=2)

# Plot networks
nx.draw_networkx(G, pos,
                 font_size=16,
                 width=3,
                 edge_color='blue',
                 node_color='red',
                 with_labels = False,
                  ax=ax)
# Create offset labels
```

```
for key, value in pos.items():
    x, y = value[0]+.135, value[1]+.045
    ax.text(x, y,
            s=key,
            #bbox=dict(facecolor='red', alpha=0.25),
            horizontalalignment='center', fontsize=12)

plt.show()
```

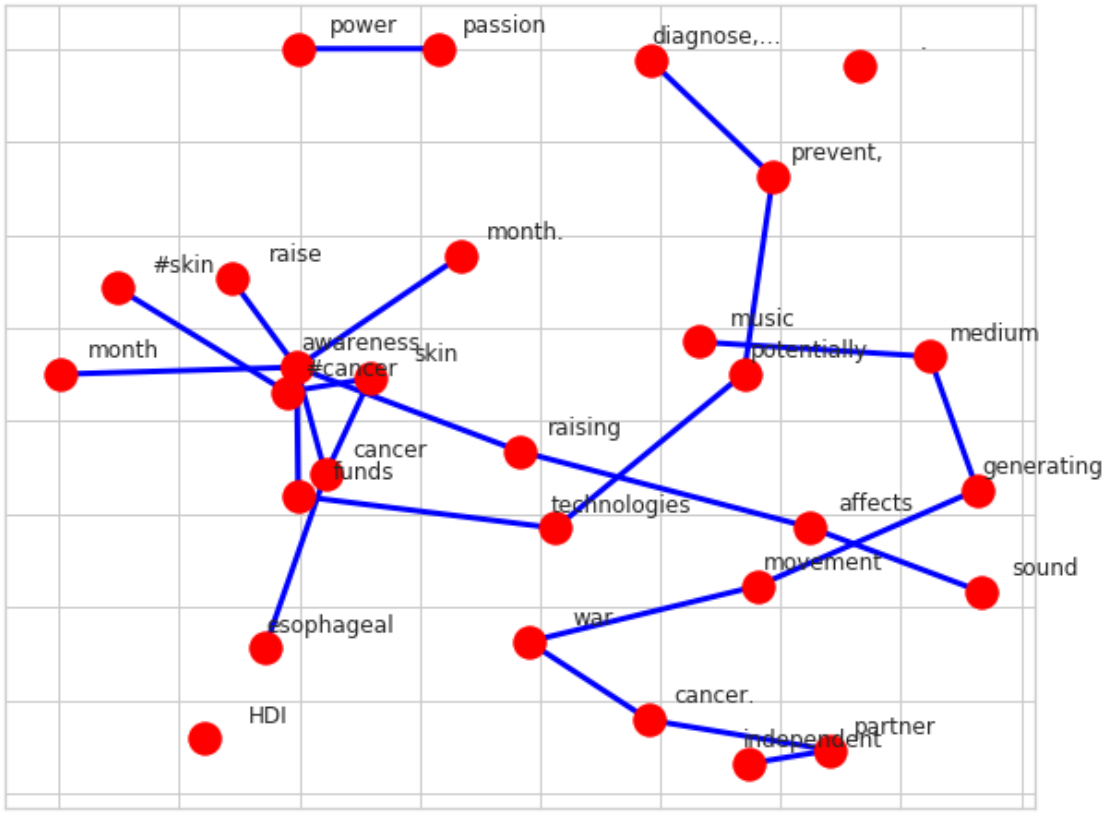


[108]:
```
# Create network plot
H = nx.Graph()
```

[109]:
```
# Create connections between nodes
for k, v in d[0].items():
    H.add_edge(k[0], k[1], weight=(v * 10))

H.add_node("GDP", weight=100)
```

[110]:
```
fig, ax = plt.subplots(figsize=(10, 8))

pos = nx.spring_layout(G, k=2)
```

```
# Plot networks
nx.draw_networkx(G, pos,
                 font_size=16,
                 width=3,
                 edge_color='blue',
                 node_color='red',
                 with_labels = False,
                  ax=ax)
# Create offset labels
for key, value in pos.items():
    x, y = value[0]+.135, value[1]+.045
    ax.text(x, y,
            s=key,
            #bbox=dict(facecolor='red', alpha=0.25),
            horizontalalignment='center', fontsize=12)

plt.show()
```



[ ]:

[ ]:
```
#Sentiment analysis
```

[111]:
```
auth = tw.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tw.API(auth, wait_on_rate_limit=True)
```

[112]:
```
def remove_url(txt):
    """Replace URLs found in a text string with nothing
    (i.e. it will remove the URL from the string).

    Parameters
    ----------
    txt : string
        A text string that you want to parse and remove urls.

    Returns
    -------
    The same txt string with url's removed.
    """

    return " ".join(re.sub("([^0-9A-Za-z \t])|(\w+:\/\/\S+)", "", txt).split())
```

[113]:
```
# Create a custom search term and define the number of tweets
search_term = "#cancer+awareness -filter:retweets"

tweets = tw.Cursor(api.search,
                   q=search_term,
                   lang="en",
                   since='2018-11-01').items(1000)

# Remove URLs
tweets_no_urls = [remove_url(tweet.text) for tweet in tweets]
```

[115]:
```
from textblob import TextBlob
```

[116]:
```
# Create textblob objects of the tweets
sentiment_objects = [TextBlob(tweet) for tweet in tweets_no_urls]

sentiment_objects[0].polarity, sentiment_objects[0]
```

[116]:
```
(0.0,
 TextBlob("With the support of emerging musicians we bring the power and passion
of people and music together to raise funds"))
```

[117]:
```
# Create list of polarity valuesx and tweet text
sentiment_values = [[tweet.sentiment.polarity, str(tweet)] for tweet in␣
 ↪sentiment_objects]
```

```
sentiment_values[0]
```

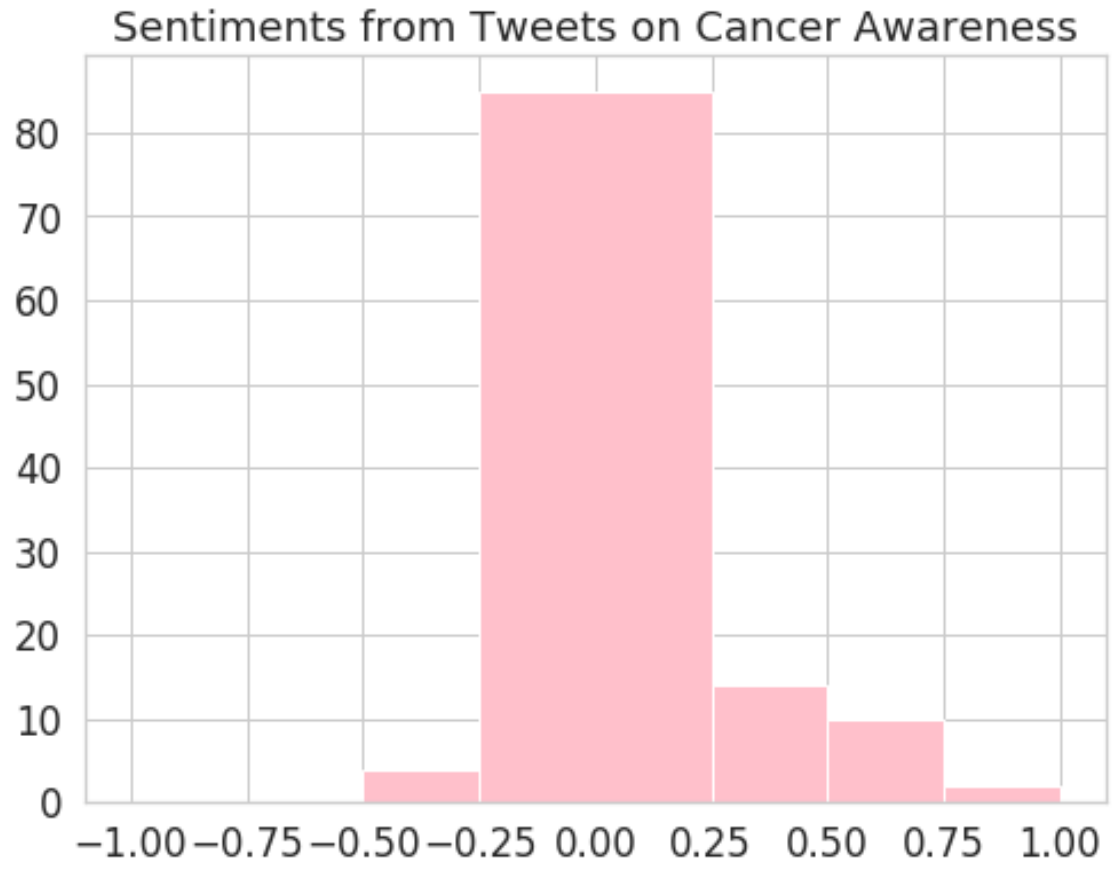
[117]:
```
[0.0,
 'With the support of emerging musicians we bring the power and passion of
people and music together to raise funds']
```

[118]:
```
# Create dataframe containing the polarity value and tweet text
sentiment_df = pd.DataFrame(sentiment_values, columns=["polarity", "tweet"])

sentiment_df.head()
```

[118]:

	polarity	tweet
0	0.0	With the support of emerging musicians we brin…
1	0.0	Did you know You might wanna research a bit wa…
2	0.1	Skin Cancer is Americas most common cancer wit…
3	0.2	What you may not know about breast cancer and …
4	0.0	QampA Determining foods to avoid when managing…

[119]:
```
fig, ax = plt.subplots(figsize=(8, 6))

# Plot histogram of the polarity values
sentiment_df.hist(bins=[-1, -0.75, -0.5, -0.25, 0.25, 0.5, 0.75, 1],
             ax=ax,
             color="pink")

plt.title("Sentiments from Tweets on Cancer Awareness")
plt.show()
```


Sentiments from Tweets on Cancer Awareness
80
70
60
50
40
30
20
10
0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

[ ]: