News Vs. Tweets

Konain Niaz

Abstract

I used a Pakistani News Headlines dataset and the Twitter API, to investigate the vocabulary used in both mediums.

I wanted to see whether major news networks have an effect on the topics and/or debates regarding women that occur online.

More specifically I wanted to see if certain words repeated themselves online after appearing on major news networks and vice versa.

Motivation

The annual Aurat (Woman's) March happens in Pakistan on 8th March. This year, in 2022, I attended the march for the first time.

I noticed that the coverage of the march by major news networks was different from the coverage of the march on social media; the news headlines only stated facts, while the posts on social media discussed the need for more public spaces involving women and their effects - good or bad - on society.

I wanted to see whether the major news headlines guided the debates on social media by the vocabulary they use.

Dataset(s)

I used a Pakistan News Headlines dataset from opendata.com.pk. It had 27,000 rows.

I also used the Twitter API to retrieve tweets related to AuratMarch2022 and saved that as a dataset with 2000 rows (including retweets).

Data Preparation and Cleaning

From the news headlines dataset, I filtered the stories to only those that contained the words: women, woman, girl, female, she, and her. Then I created a corpus by putting each positive story in one folder and each negative story in another. There were 363 negatives, and 235 positives.

I did the same with the twitter dataset and created a corpus with 185 negative tweets and 195 positive tweets.

Notes on the labels:
- Any news relating to a police investigation of a crime against women, was labeled negative. Even if the story excerpt shows the investigation is making progress.
- News of a woman speaking out against misogyny or oppression was labeled negative because the vocabulary in the text is about the incident, which is negative. Even though the news itself might be positive in context.

Research Question(s)

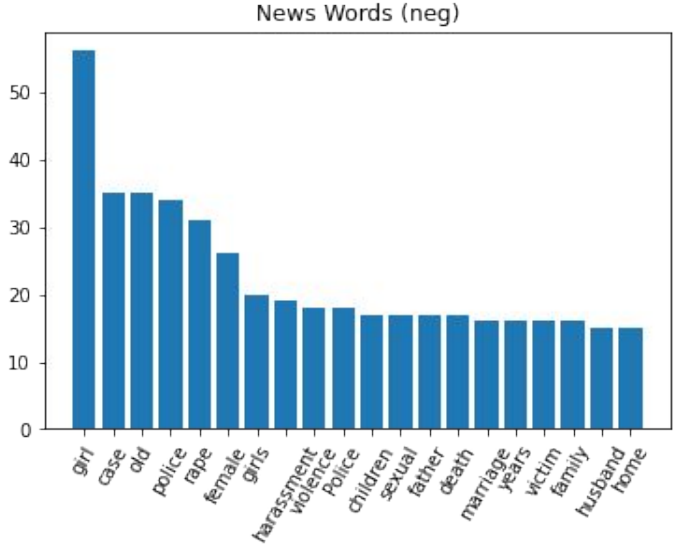
Do news headlines concerning women and tweets about the women’s march, share similar vocabulary?

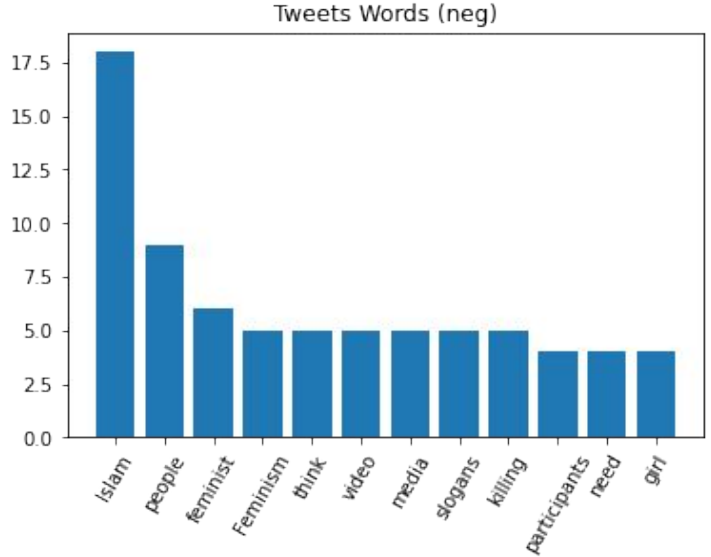
Methods

After creating corpora of the news headlines and the tweets, I created counter objects to get a list of all the words used in both fileids.

Using the most_common() method, I created lists of positive and negative features for both the news headlines and tweets. I created dataframes from the lists and plotted them for comparison.

Findings, Negative Words



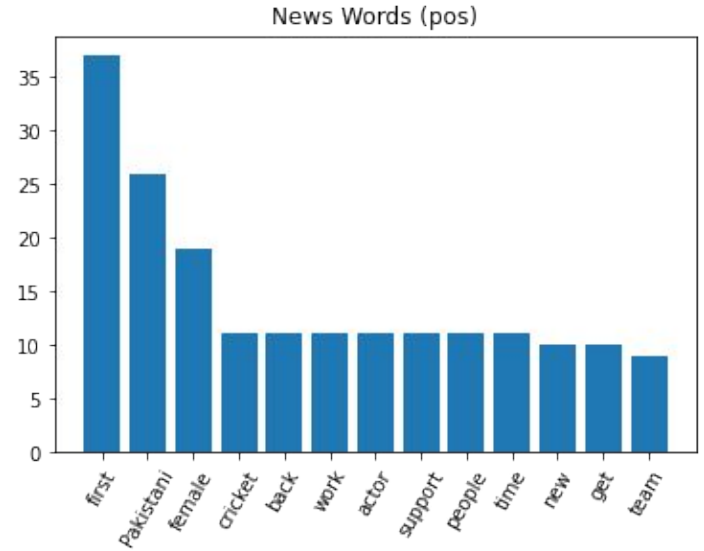


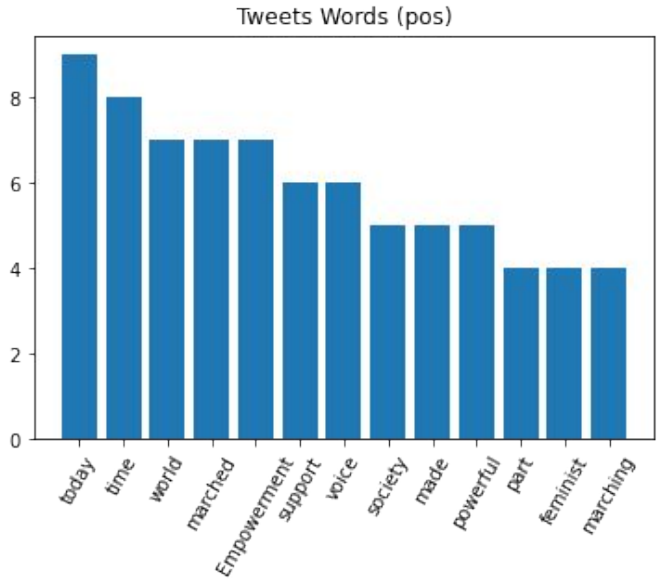
From the News words we learn that the negative news coverage is mostly about sexual assualt cases against girls.

The Tweets words show that the negative posts are mostly related to ‘islam’ and it’s ideas about women and organization of people and their rights. These words appear independently from the news words. But words like ‘media’, ‘killing’, and ‘girl’ intersect.

This shows that while the topics on twitter are mostly independent, there is an intersection on the topics of violence on girls. And that the word ‘girl’ has become synonymous with ‘violence’ and ‘media’ for the twitter users and the major news networks.

Findings, Positive Words





The News words show that the positive news coverage is mostly about a ‘first’ ‘female’ feat in some domain. Or stories about actors going back to work. Or stories about the pakistani female cricket team.

The Tweets words show that the positive posts are mostly related to the ‘rights’ that are being demanded ‘today’ at the march. Most of the words appear independently here that we have not seen before. Showing that the “positive” twitter posts are very different from the major news headlines, in their topics and vocabulary.

Notice that the world ‘girl’ does not appear in either sets here.

Conclusions

For the most part, the news headlines seemed to be far off from the debates taking place on social media and the concerns of the users. The positive news headlines were an incomplete depiction of positive scenarios for women that the online community discussed. And the negative news focused only on violence against women while ignoring religious concerns of some people.

I created sentiment classifiers with both datasets and ran them on their test sets. The News data classifier predicted on its own test set with an accuracy of 75.2%.
The Twitter data classifier predicted on its own test set with an accuracy of 82%.

I ran the News classifier on the Twitter data test set, it predicted with an accuracy of 62.8%.
But, when I ran the Twitter classifier on the News data test set, and it predicted with an accuracy of 53.7%.

This shows that Twitter contains most of the vocabulary of the News but the News does not contain most of the vocabulary on Twitter.

Limitations

One large limitation was the language barrier. English is a secondary language in Pakistan. For a more in depth sentiment analysis the online Urdu community would have to be analyzed, as they inhabit a much larger public space.

Another limitation was my personal bias when labeling the News and Tweets as positive or negative.

Acknowledgements

I got my data from opendata.com.pk. It is a free online resource for datasets related to Pakistan. The website was very handy when I was doing my preliminary research.

I would also like to acknowledge the Aurat March organizers for hosting the march and empowering so many Pakistanis to reclaim public space and think about research questions like the ones discussed here.

References

I did all of the work on my own.

In [1]:
```
import pandas as pd
import numpy as np
import math as math
import nltk
import tweepy
import string
import csv
import sklearn
import matplotlib.pyplot as plt
from collections import Counter
from nltk.corpus import CategorizedPlaintextCorpusReader
from nltk.corpus import LazyCorpusLoader
from nltk.classify import NaiveBayesClassifier
from nltk.tokenize import word_tokenize
```

Creating bag of words that appear too many times and must be filtered for better analysis

In [2]:
```
recurring = ['rights','Stop','year','Day','would','Khan','She','said','still','nothing','r
             'In','—£','âœ','Of','So','AuratMarch2022â','AuratAzadiMarch',
             'Ø','It','AuratMarchKHI','They','How','A','men','This','amp',
             'Aurat','AuratAzadiJalsa2022','If','What','The','à','womenâ',
             ',"','€˜','€™','€¦' , 'https', '://', 'co','RT','AuratMarch2022',
             'women','ðŸ','â','€¦"','March', 'AuratMarch','I','woman','march',
             'Women','Woman','±','¦','€','€\x9d','""','›``']
```

In [3]:
```
nltk.corpus.stopwords.words("english")


useless_words = (nltk.corpus.stopwords.words("english") +
                 list(string.punctuation) +
                 recurring
                )
```

In [4]:
```
def bag_of_words(words):
    return { word:True for word in words if not word in useless_words }
```

In [5]:
```
#Setting up the keys and tokens
c_k = "1kivB3odKErhRTM4OA8vdeTWB"
c_s = "svMXMfWGtW57IgGlMcoqhWiL2isJxcZNT9sTxnOFYRq1ZZr21q"

a_t = "1100528374102007811-DXeGxfEEMcscoamygqM4eMhtRJVQ8C"
a_s = "Ri1rZ4OEzNz8DvgznXIBosHLgdOOzagTr1kFbDT9ntOUT"

auth = tweepy.OAuthHandler(c_k, c_s)
auth.set_access_token(a_t, a_s)
api = tweepy.API(auth)

# Nothing to see by displaying twitter_api except that it's now a
# defined variable

print(api)
```

```
<tweepy.api.API object at 0x00000245861154F0>
```

In [6]:
```
Lahore_ID = 2211177
```

In [7]:

```
search_results = tweepy.Cursor(api.search_tweets, q = "#AuratMarch2020", lang='en').items
```

In [8]:
```
tweets=[]
retweet_count = []

for tweet in search_results:
    tweets.append(tweet.text)
    retweet_count.append(tweet.retweet_count)
```

In [9]:
```
df = pd.DataFrame({'Tweet':tweets, 'Retweet Count':retweet_count})
```

In [10]:
```
df_sorted = df.sort_values(by='Retweet Count', ascending=0)
```

Saving the tweets as a csv file so I can label each tweet as pos or neg

In [11]:
```
df_sorted.to_csv('auratmarch.csv')
```

Created a folder titled amarch_tweets with subfolders of neg and pos tweets

In [12]:
```
amarch_tweets = None
amarch_tweets = CategorizedPlaintextCorpusReader(
    r'./Downloads/amarch_tweets',
    r'(?!\.).*\.txt',
    cat_pattern=r'(neg|pos)/.*',
    encoding="utf-8"
)
```

In [13]:
```
amarch_tweets.categories()
```

Out[13]:
```
['neg', 'pos']
```

In [14]:
```
amarch_tweets.fileids()[-5:]
```

Out[14]:
```
['pos/95.txt', 'pos/96.txt', 'pos/97.txt', 'pos/98.txt', 'pos/99.txt']
```

In [15]:
```
print(amarch_tweets.raw(fileids=amarch_tweets.fileids('neg')[5]))
```

```
Unfortunately, We Failed to Teach Young Generation the rights of Women's in Islam, what ou
r Holy Prophet P.B.U.H Teâ€¦ https://t.co/zktuCS7muJ
```

In [16]:
```
amarch_tweets.words(fileids=amarch_tweets.fileids('neg')[5])
```

Out[16]:
```
['Unfortunately', ',', 'We', 'Failed', 'to', 'Teach', ...]
```

In [17]:
```
pos_tweet_words = [word for word in amarch_tweets.words(categories=['pos'])
             if not word in useless_words]

pos_tweet_counter = Counter(pos_tweet_words)

neg_tweet_words = [word for word in amarch_tweets.words(categories=['neg'])
             if not word in useless_words]
```

```
neg_tweet_counter = Counter(neg_tweet_words)
```

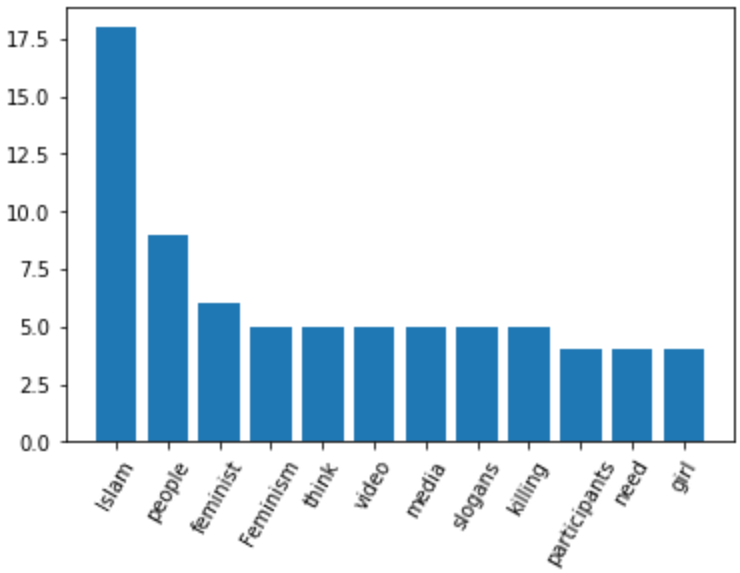
In [18]:
```
neg_tweet_top = neg_tweet_counter.most_common()[:12]
pos_tweet_top = pos_tweet_counter.most_common()[:13]
```

In [19]:
```
df_neg_tweet = pd.DataFrame(neg_tweet_top, columns=['Words','Times Appeared'])
df_pos_tweet = pd.DataFrame(pos_tweet_top, columns=['Words','Times Appeared'])
df_neg_tweet
```

Out[19]:

	Words	Times Appeared
0	Islam	18
1	people	9
2	feminist	6
3	Feminism	5
4	think	5
5	video	5
6	media	5
7	slogans	5
8	killing	5
9	participants	4
10	need	4
11	girl	4

In [20]:
```
plt.bar(df_neg_tweet['Words'], df_neg_tweet['Times Appeared'].values)

plt.xticks(rotation=60)

plt.show()
```



Download the News Headlines Dataset

In [21]:
```
news_data = pd.read_csv('./Downloads/paknews/News_Headlines.csv')
```

In [22]:
```
news_data = news_data.dropna()
```

In [23]:
```
contains_woman = news_data[news_data['Story Excerpt'].str.contains('woman')
                           | news_data['Story Excerpt'].str.contains('girl')
                           | news_data['Story Excerpt'].str.contains('women')
                           | news_data['Story Excerpt'].str.contains('female')
                           | news_data['Story Excerpt'].str.contains(' her ')
                           | news_data['Story Excerpt'].str.contains(' she ')]
```

Save this as a seperate csv file for labeling.

In [24]:
```
contains_woman.to_csv('news_woman.csv')
```

In [25]:
```
news_headlines = None
news_headlines = CategorizedPlaintextCorpusReader(
    r'./Downloads/news_headlines',
    r'(?!\.).*\.txt',
    cat_pattern=r'(neg|pos)/.*',
    encoding="utf-8"
)
```

In [26]:
```
news_headlines.categories()
```

Out[26]:
```
['neg', 'pos']
```

In [27]:
```
news_headlines.fileids()[:5]
```

Out[27]:
```
['neg/0.txt', 'neg/1.txt', 'neg/10.txt', 'neg/100.txt', 'neg/101.txt']
```

In [28]:
```
negative_fileids = news_headlines.fileids('neg')
positive_fileids = news_headlines.fileids('pos')
```

In [29]:
```
print(news_headlines.raw(fileids=negative_fileids[4]))
```

```
Education official seeks action against male teachers over â€ħarassmentâ€™ in Kohat     Sa
ys they were putting pressure on her through their cronies in the education department and
in political circles.
```

In [30]:
```
news_headlines.words(fileids=negative_fileids[4])
```

Out[30]:
```
['Education', 'official', 'seeks', 'action', 'against', ...]
```

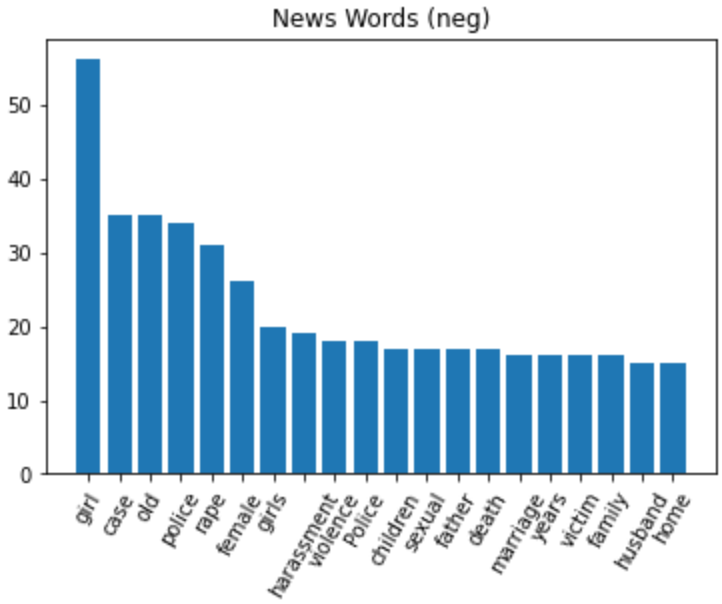
In [31]:
```
neg_words = [word for word in news_headlines.words(categories=['neg'])
             if not word in useless_words]

neg_counter = Counter(neg_words)
```

```
pos_words = [word for word in news_headlines.words(categories=['pos'])
             if not word in useless_words]

pos_counter = Counter(pos_words)
```

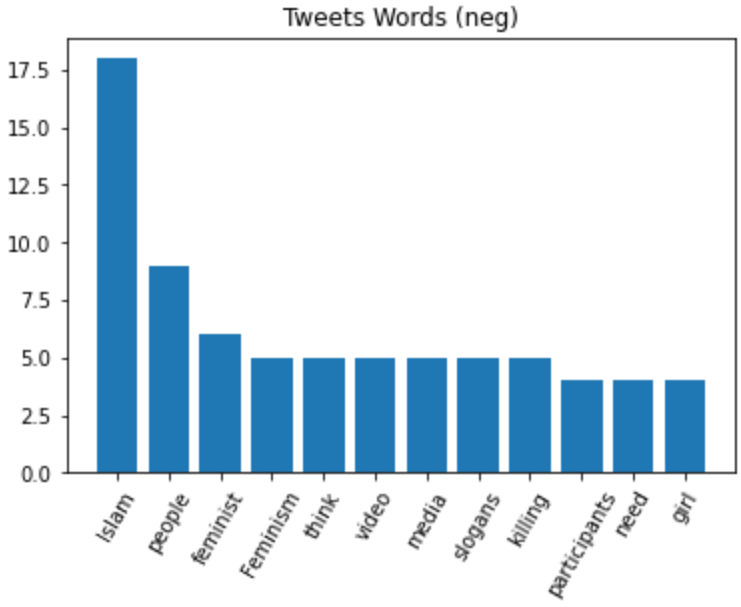
In [32]:
```
neg_top = neg_counter.most_common()[:20]
pos_top = pos_counter.most_common()[:13]
```

In [33]:
```
df_neg = pd.DataFrame(neg_top, columns=['Words','Times Appeared'])
df_pos = pd.DataFrame(pos_top, columns=['Words','Times Appeared'])
```

In [34]:
```
plt.bar(df_neg['Words'], df_neg['Times Appeared'].values)

plt.xticks(rotation=60)

plt.title('News Words (neg)')

plt.show()
```

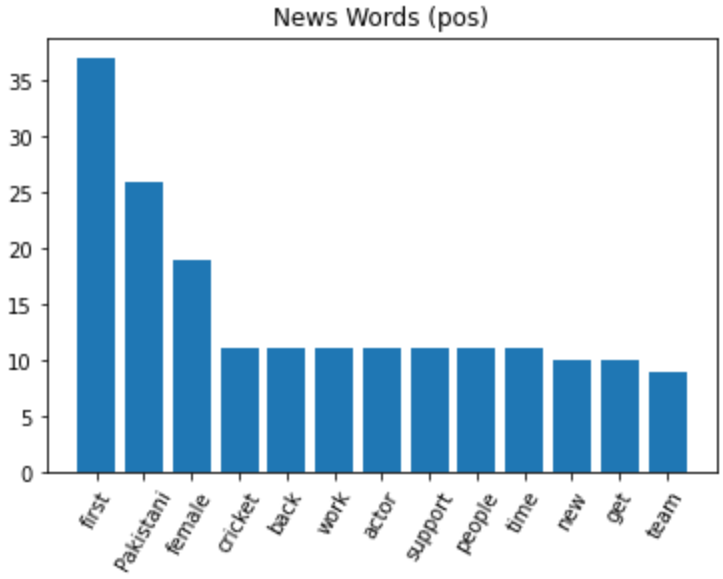


In [35]:
```
plt.bar(df_neg_tweet['Words'], df_neg_tweet['Times Appeared'].values)

plt.xticks(rotation=60)

plt.title('Tweets Words (neg)')
plt.show()
```



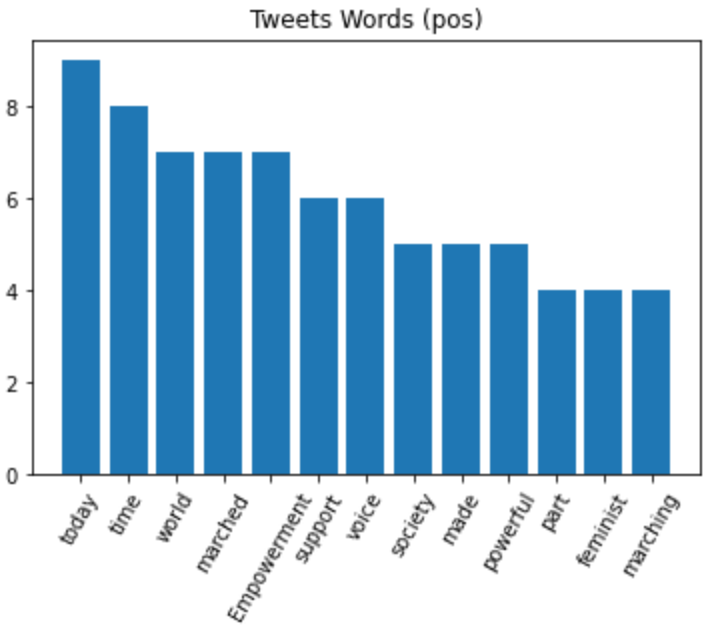
In [36]:

```
plt.bar(df_pos['Words'], df_pos['Times Appeared'].values)

plt.xticks(rotation=60)

plt.title('News Words (pos)')

plt.show()
```



In [37]:

```
plt.bar(df_pos_tweet['Words'], df_pos_tweet['Times Appeared'].values)

plt.xticks(rotation=60)

plt.title('Tweets Words (pos)')

plt.show()
```



Creating sentiment classifiers

In [38]:
```
negative_features = [
    (bag_of_words(news_headlines.words(fileids=[i])),
    'neg')
    for i in negative_fileids
]

positive_features = [
    (bag_of_words(news_headlines.words(fileids=[i])),
     'pos '
    )
    for i in positive_fileids
]
```

In [39]:
```
split_neg = int(len(negative_fileids)*0.8)
split_pos = int(len(positive_fileids)*0.8)
```

In [40]:
```
classifier = NaiveBayesClassifier.train(positive_features[:split_pos]+negative_features[:s
```

In [41]:
```
nltk.classify.util.accuracy(
    classifier,
    positive_features[:split_pos]+negative_features[:split_neg])*100
```

Out[41]:
```
98.74739039665971
```

In [42]:
```
nltk.classify.util.accuracy(
    classifier,
    positive_features[split_pos:]+negative_features[split_neg:])*100
```

Out[42]:
```
75.20661157024794
```

Creating lists of features from the twitter dataset

In [43]:
```
neg_tweet_fileids = amarch_tweets.fileids('neg')
```

```
pos_tweet_fileids = amarch_tweets.fileids('pos')
```

In [44]:
```
neg_tweet_features = [
    (bag_of_words(amarch_tweets.words(fileids=[i])),
    'neg')
    for i in neg_tweet_fileids
]

pos_tweet_features = [
    (bag_of_words(amarch_tweets.words(fileids=[i])),
     'pos '
    )
    for i in pos_tweet_fileids
]
```

Running the News Classifier on the Twitter dataset

In [51]:
```
nltk.classify.util.accuracy(
    classifier,
   pos_tweet_features[split_tw_pos:]+neg_tweet_features[split_tw_neg:])*100
```

Out[51]:
```
62.82051282051282
```

Creating another classifier with the twitter dataset

In [46]:
```
split_tw_neg = int(len(neg_tweet_fileids)*0.8)
split_tw_pos = int(len(pos_tweet_fileids)*0.8)
```

In [47]:
```
tweet_classifier = NaiveBayesClassifier.train(pos_tweet_features[:split_pos]+neg_tweet_fea
```

In [48]:
```
nltk.classify.util.accuracy(
    tweet_classifier,
    pos_tweet_features[:split_tw_pos]+neg_tweet_features[:split_tw_neg])*100
```

Out[48]:
```
97.39413680781759
```

In [50]:
```
nltk.classify.util.accuracy(
    tweet_classifier,
    pos_tweet_features[split_tw_pos:]+neg_tweet_features[split_tw_neg:])*100
```

Out[50]:
```
82.05128205128204
```

Running the Twitter Classifier on the News dataset features

In [52]:
```
nltk.classify.util.accuracy(
    tweet_classifier,
    positive_features[split_pos:]+negative_features[split_neg:])*100
```

Out[52]:
```
53.71900826446281
```