Christina Holman
ENGR 2510 - Steele
4 October 2016

Text Mining: The Twitter Personalities of the Monsters of *Sesame Street*

I decided to examine a series of tweets supposedly written by Sesame Street's famous, lovable character "Elmo." I have grabbed a maximum of 200 tweets into text files in order to perform word frequency and sentiment analysis. I developed frequency histograms to show Elmo's top twenty words. I also compared Elmo's tweets with those of Oscar the Grouch — as well as Cookie Monster. I developed histograms of their twenty, most used words too.

Implementation:

Using the tweedy package from Twitter, I pulled tweets (excluding retweets) from Elmo's, Oscar's, and Cookie's timelines and, after stripping them of their ID and author information, I emptied solely the text into a txt file. Then, the words in the file were separated into a list called *outtweets* for each monster. For word frequency analysis, I created a python script called *toptwenty.py:* the text files were all joined in an array of monsters. For each monster, the code creates a dictionary, where each word is attached the value of its frequency. From the histogram dictionary, I create a sorted list of the word-value combinations and print twenty words organized by the value (*key)*. Using matplotlib, I create a histogram of the twenty most frequent words.

Sentiment analysis uses two text files — one of 'positive' words and the other of 'negative' words — to count the presence of positives versus that of negative in addition to a function calculating a monster's polarity towards positivity or negativity. I chose to include the positive and negative word counts in order to provide some basis for the *sentiment* function: however, there is a difference in the number of tweets available as Oscar tweeted much less than Elmo and Oscar. The script also uses cosine similarity to plot the similarity or dissimilarity of the texts listed.

Results:

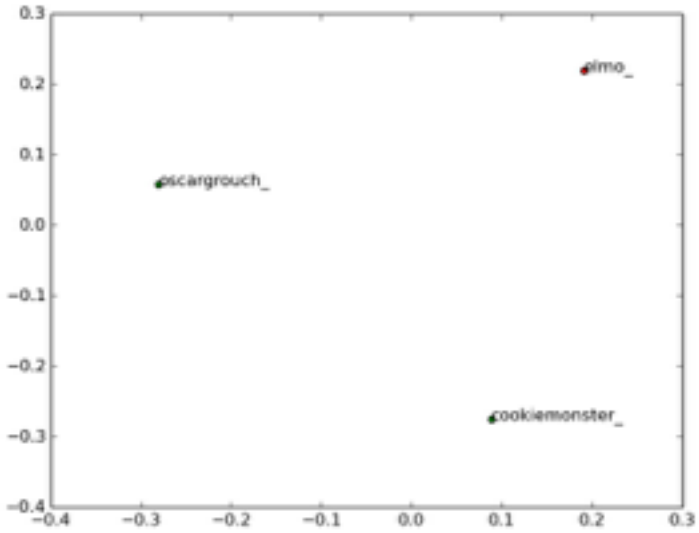
It is no surprise that Elmo is the most positive, energetic monster in the group — and that Oscar the Grouch is the negative. However, I was intrigued by the positivity calculation of Cookie Monster as well as the subjectivity of his tweets:

Elmo: (0.3963483989630115, 0.5356215965705012);
Oscar: (0.16815814393939393, 0.38257575757575757);
Cookie: (0.3104059709413679, 0.5874839248971193)

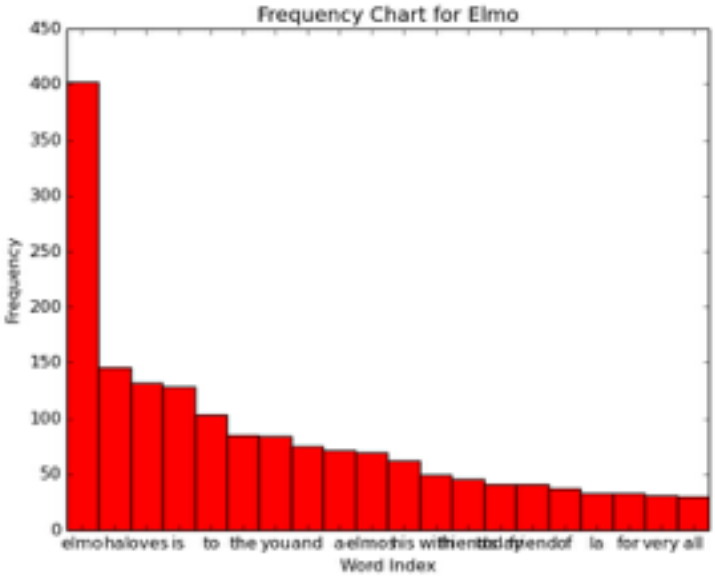
The script printed the cosine similarity matrix as:

```
[[ 0.   0.5  0.5]
 [ 0.5  0.   0.5]
 [ 0.5  0.5  0. ]]
```

The matrix shows the texts are almost equidistant from one another, and, in fact we calculating the distance between any of two points, the distance equals .5. Looking at the plot of the twitter personalities, they are all equally different from one another. However, the frequency histograms for each could not be more different in value and word choice.

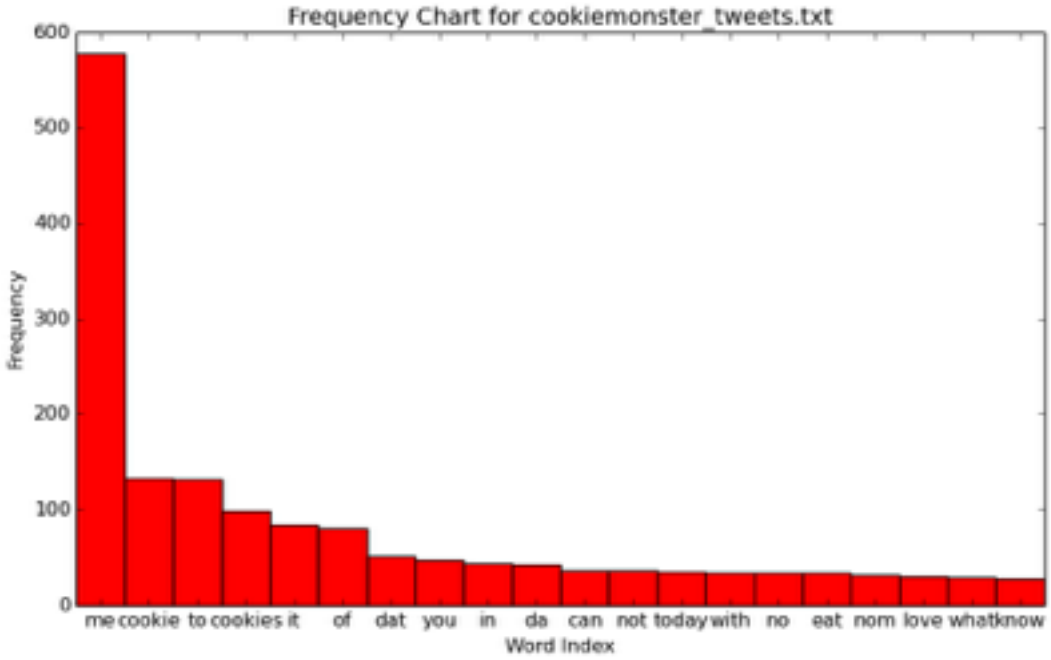


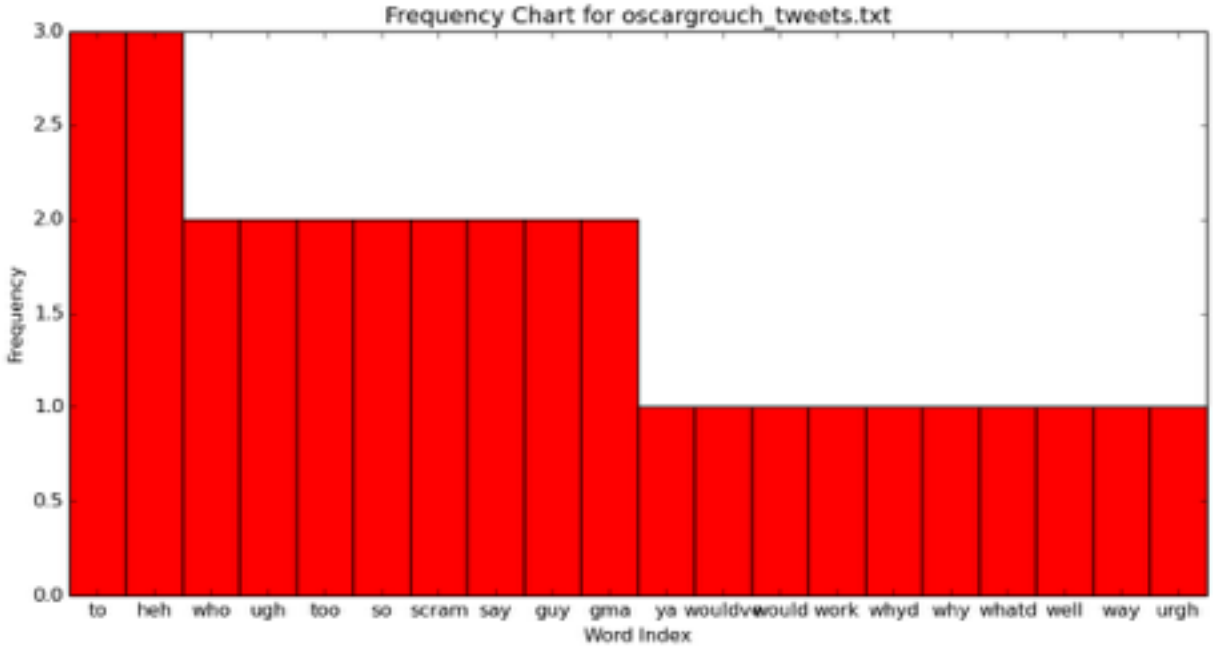
Elmo enjoys speaking in third person, laughing and expressing his love for other people. Conversely, Cookie Monster (below) expresses his love specifically for cookies and refers to himself often in first person possessive ("me"). Oscar the Grouch (below) is more likely to tell someone to 'scram,' express his disdain ('ugh' or hey), and ask sarcastic questions.



Reflection:

I enjoyed the exercise as I could be creative with it and see the power of word analysis through code. However, my project took longer than I planned as I tried to push beyond what I knew how to do in order to do what I wanted. Although tedious, I think I've improved my debugging skills quite a bit and have learned to organize my code better. If I had to the assignment again, I would have looked more for shortcuts and packages already in existence, rather than trying to re-invent the wheel. I would also add more *Sesame Street* characters like Big Bird, Count von Count, or Grover .


Frequency Chart for cookiemonster_tweets.txt
Frequency
Word Index
0
100
200
300
400
500
600
me
cookie
to
cookies
it
of
dat
you
in
da
can
not
today
with
no
eat
nom
love
what
know


Frequency Chart for oscargrouch_tweets.txt
Frequency
Word Index
0.0
0.5
1.0
1.5
2.0
2.5
3.0
to
heh
who
ugh
too
so
scram
say
guy
gma
ya
would
work
whyd
why
whatd
well
way
urgh