Alexander Hoppe
SoftDes Spring 16
Text Mining Reflection

Overview:

For my text mining, I used my Facebook data dump (which is surprisingly easy to get) as my data source. This dump included my entire facebook message history, and I wanted to extract the Class of 2019 group chat out of that, and analyze the messages we send. I originally hoped to create a word map out of the list of words, however, I ended up doing frequency analysis.

Implementation:

My implementation was not incredibly complex. I used the BeautifulSoup and wordcloud modules (although a little unsuccessfully with the wordcloud) and I scraped my Facebook messages in four steps. My first algorithmic problem was using BeautifulSoup to search through a massive nested tree of message threads for the thread specified by the input string. Since the messages are not sorted (at least as far as I could tell) I intended used a linear search to navigate the HTML tree until the right message thread was encountered. Unfortunately, this search process was not reliable enough, so it remains unimplemented.

The rest of my project was text processing. My next step was to pull all of the messages out of the message thread block and trim them of HTML tags, which I did with list comprehensions. Next, I tried to make a wordcloud with the resulting message text, but the module was uncooperative. After that, I used dictionaries to first make a histogram of all of the words in our group chat, and then used tuples and a Decorate Sort Undecorate pattern to sort them in frequency order. This was a design decision I made because I preferred to use the convenient tuple sorting behavior over messing around with an OrderedDict.

I also made my program command-line friendly. It prompts the user for input on the number of words to display, as well as the thread tag it is looking for in the messages.htm file it expects in the working directory. This is simply a list of all the names in the Facebook group conversation, which is easiest to just find in the document and paste into the command line, unfortunately.

Results:

Since I did frequency analysis, I read through the list of words and it is surprisingly very characteristic of our group conversations. Of the names that get mentioned, Sam's name is the first, and such words like toppings and pizza happen quite frequently. This is pretty humorous, as it reflects one of the major functions of our group chat, ordering pizza! We also get words like 'Olin' that come up over 300 times out of the 45,000 words we've written to one another. Unsurprisingly, short words like 'I' and 'the' are the most common, but neither of these is more than 2% of the words we use.

The list is interesting because it does not look like normal writing. In fact, reading it, it becomes clear that these are short messages that are usually asking if “anyone (320)” has done something or “know (205)” something or saying “hey (125)” “you (779)” or “do (226)” “we (230)” “want (128)” “this (205)” or something along those lines. These are all messages pulled from the top 50 words in our chat, which can be found at the end of this document.

Reflection

From a process point of view, this was a terribly executed project. I had trouble finding something I was motivated about analyzing, and I let other things take priority over it, so I ended up doing analysis that wasn't very compelling with data that had so much more potential over the course of the final day before the project was due. I think it was under-scoped, and I fully intend to create something cool with this dataset in the coming days, like a program that auto-generates messages that could belong in our Facebook chat using Markov analysis. From a text-mining learning perspective, I familiarized myself with Beautiful Soup more than I had at WHACK at Wellesley, which is useful, but I also gained a healthy respect for web scraping programs. It's really difficult and unwieldy. I also learned a bit about file I/O and making command-line interfaces in Python. My unit testing was limited, because BeautifulSoup uses its own funky types that were making it a problem. Overall, I could've done better, and I wanted to, but I mismanaged my time.

Frequency Analysis on C/O 2019 Chat:
Of 44712 total words in the chat,
the 50 most used words in our chat are:

```
(1303, 'the')
(1096, 'i')
(1002, 'to')
(839, 'a')
(779, 'you')
(659, 'in')
(652, 'and')
(637, 'is')
(586, 'it')
(487, 'of')
(441, 'for')
(398, 'that')
(332, 'have')
(326, 'on')
(320, 'anyone')
(301, 'if')
(300, '')
(293, 'my')
(289, 'can')
(287, 'so')
(279, 'at')
(270, 'me')
(267, 'but')
(258, 'be')
(257, "i'm")
(251, 'just')
(245, "it's")
(237, 'with')
(230, 'we')
(230, 'not')
(226, 'do')
(209, 'what')
(205, 'this')
(205, 'know')
(194, 'get')
(193, 'are')
(186, 'was')
(175, 'no')
(169, 'like')
(165, 'or')
(160, 'now')
(160, 'all')
(154, 'up')
(147, 'your')
(138, 'guys')
(137, 'one')
```

```
(128, 'want')
(128, 'about')
(125, 'where')
(125, 'hey')
```