Noah Chapin & Julia Chon

Professor Li

MIS3640-01

21 October 2020

Assignment 2 Reflection

1. **Project Overview [Maximum 100 words] What data source(s) did you use and what technique(s) did you use analyze/process them? What did you hope to learn/create?**

The increasingly chaotic nature of political media generated from the upcoming presidential election has garnered our interest in analyzing the tweets about President Donald Trump. Thus, we decided to designate Twitter as the sole data source for the project.

For this project, we wrote a program to search for tweets containing the term "Donald Trump," and analyzed the word frequencies to create a bar graph depicting the top 10 most frequently used words, and a word cloud of up to 1,000 of the most frequently used words. The more frequently used the word is, the larger the depiction.

2. **Implementation [~2-3 paragraphs] Describe your implementation at a system architecture level. You should NOT walk through your code line by line, or explain every function (we can get that from your docstrings). Instead, talk about the major components, algorithms, data structures and how they fit together. You should also discuss at least one design decision where you had to choose between multiple alternatives, and explain why you made the choice you did.**

The initial candidate when deciding how to access the tweets was the Twython library. After further development, it was revealed that tweets with more than 140 characters were truncated after Twitter launched the "extended_tweet" field to accommodate the 280 character tweets in 2017. V2 Twitter API enables users to obtain the entire tweet, though multiple attempts and in-depth research found that Twython does not support the v2 Twitter API. Tweepy was briefly used until it was concluded that accessing the full text added another level of complexity that would not be sustainable provided the amount of restructuring necessary to make it work. Thus, Twython remained the optimal choice.

The raw data that Twython returned was structured as a dictionary as it is the most efficient way of reaching the tweet text. This structure served two key purposes. First, iterating through the dictionary allowed for the text to be cleaned and reformatted. Second, it positioned the tweet text after being properly formatted to be translated into a list for further processing. The list was crucial in removing the majority of outliers, especially considering how the tweets

were already truncated. In addition, the list made the task of filtering out the basic stop words, along with other case-based language that would distort the analysis (i.e. "trump", "biden", "rt"). This structure was essential in identifying the frequency of the words, whereas the raw data dictionary would not be nearly as accommodating. These two components were designed in order to create the count dictionary, which is composed of tuples with word frequency pairs. The count dictionary serves as the foundation for the following data structures.

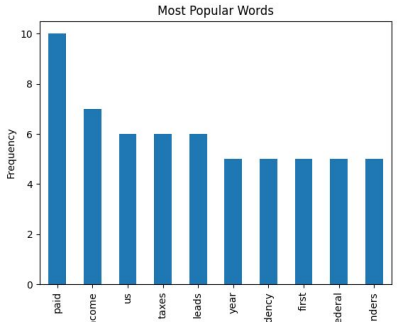
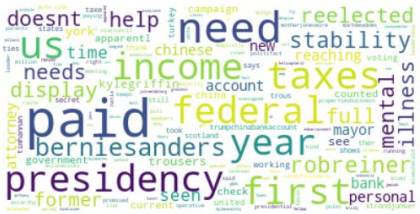
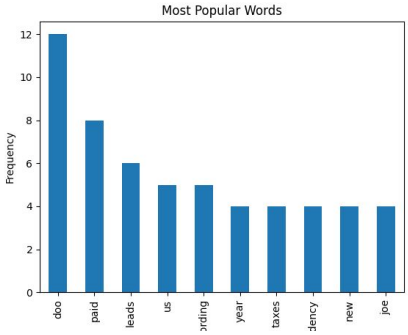
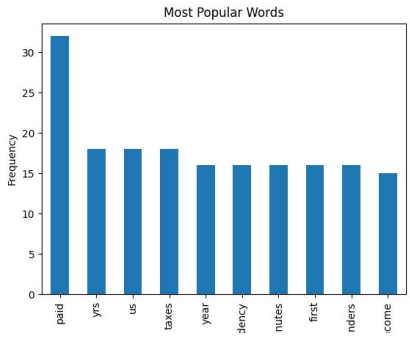
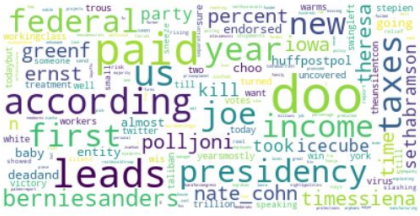
The count dictionary tuples were constructed to allow for the objects to be read and added in a seamless transition to the top list structure. The top list provides a simplified structure that is the foundation for pandalist, which slightly differs from the standpoint that it reversed the key:value pairs and contains a specified amount of the highest frequency words for the panda bar chart visualization. The top list is also the groundwork for the reversetop list structure in a similar manner. The reversetop list switches the key:value pair and converts it to a dictionary so that it may be used to generate a visual representation based upon word frequencies. While the swapping may seem to be bulkier than necessary, it has been evaluated to be more effective than translating the count dictionary to a list as the basis for other structures and reversing the top list. Unit testing uncovered that there were several points of error if the latter were to be implemented.

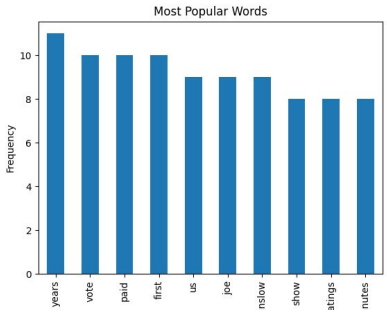
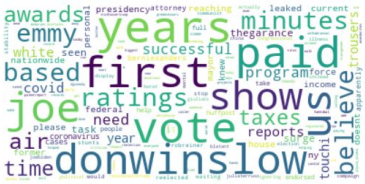
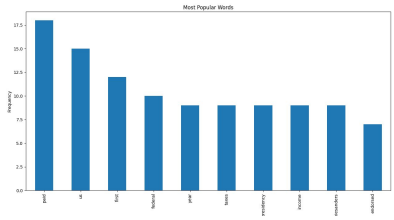
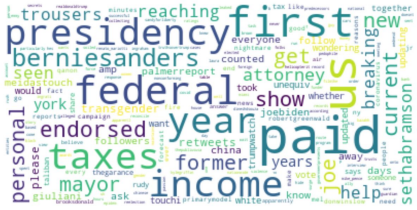
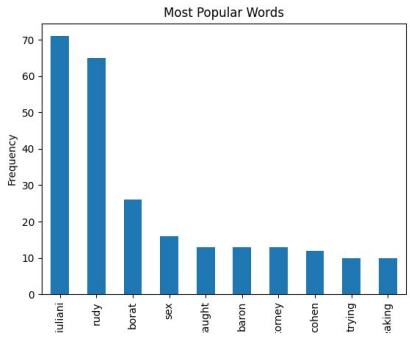
3. Results [~2-3 paragraphs + figures/examples] Present what you accomplished:

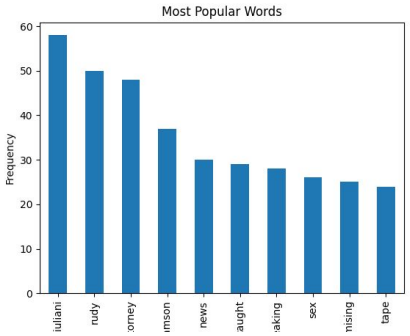
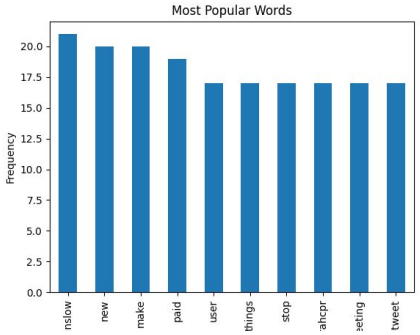
When you run the code, a bar graph of the top ten most frequently used words in the most recent tweets will appear. Afterward, a word cloud of the 1,000 most frequently used words in the most recent tweets will appear. We decided to omit several terms (rt, donald, trump, trumps, bidens, biden, republican, republicans, democrat, democrats, democratic, president) because they appeared in nearly every single tweet, and skewed the data. For example: “rt” refers to retweets, which isn’t relevant to our analysis; and “president” was omitted because nearly every tweet with “donald trump” already included “president.”

The results are different each run because the program collects the most recent tweets. (Results on following page) We realized that we could track current events and breaking news by looking at which terms were trending.

Time (with key events)	Bar	Word cloud

10/21 1:23pm News from yesterday showed that Trump **paid** more Chinese **taxes** than American taxes		
10/21 1:27pm		
10/21 1:54pm		

10/21 3:52pm Author **Don Winslow** tweeted we can't **believe** Trump because he lied about **ratings**, and the **Emmys**		
10/21 3:53pm		
10/21 4:22pm		

10/21 4:23pm **Rudy Giuliani caught** touching himself in front of **underage** girl in **Borat** movie		
10/21 5:05pm **Sarah Cooper** had two viral tweets about Trump		
6:02		

4. **Reflection [~1 paragraph] From a process point of view, what went well? What could you improve? Other possible reflection topics: Was your project appropriately scoped? Did you have a good plan for unit testing? How will you use what you learned going forward? What do you wish you knew before you started that would have helped you succeed?**

There were countless learning opportunities and successes during this project. The process planning was weak, which hindered development efforts. It was strategically illogical to dive into creating the code prior to fully researching the topics and gaining a better understanding of the system requirements. For instance, NLP was the primary pick but hours were spent attempting to fix system issues with the nltk.sentiment.vader errors. This inevitably led to many rounds of uninstalling and hunting different file paths. After correcting the problems, there was little time dedicated towards building a solid grasp of the NLP process. Needless energy was spent trying to push through the problem instead of growing the conceptual knowledge to realize what was actually required. Subsequently, the team shifted to new target analyses. It was decided that there would be a significantly greater emphasis on drawing out the scope of the project and the order in which development would take place. Unit testing became much more rigorous so that similar mistakes downstream would not pose as much of an issue as previously. The increased use of pair programming and around-the-clock availability was exactly what was needed to get the project to where it currently stands.

Also discuss your team process in your reflection. How did you plan to divide the work (e.g. split by task, always pair program together, etc.) and how did it actually happen? Were there any issues that arose while working together, and how did you address them? What would you do differently next time?

Our initial plan was to split up the coding into two parts: Noah would write the code to mine the tweets, while Julia would write the code that would perform and display the analysis and word cloud. We tried to pair program together over FaceTime over the weekend, but there was a lot of background reading and learning to do, which made pair programming tedious and time consuming.

We then decided to work apart, which meant Julia worked on researching tutorials and code to send to Noah to use, while Noah worked on the code on his own time. We FaceTimed every few days to check in with each other, and resolve any problems that came up. At the end, when we were trying to create a dataframe using pandas, we pair-programmed over WebEx, with Noah sharing control of his screen.

Programming separately and splitting by task made the process more efficient for our team for most of the assignment. If circumstances were different, and we had extra time and were in close proximity, we would try coding next to each other. Coding separately required a lot of coordination, and it took time to send and wait for replies. However, we were still successful and managed to meet our goals.

https://stackoverflow.com/questions/64001237/how-to-obtain-full-text-field-value-from-twitter-api-with-twythonstreamer

"Twython does not support v2 that was released this october"

Honor Code

“I have abided by the Babson Code of Ethics in this work and pledge to be better than that which would compromise my integrity.”

Signature: Noah Chapin
Signature: Julia Chon