Project Overview:

I chose to analyze Twitter data by searching through recorded tweets for a given query. After searching for and compiling tweets, I used a version of Markov analysis to analyze the frequency of words and the words they are followed by. I then created a function that would generate strings of words based on the Markov data.

Implementation:

For finding tweets, I simply used pattern's Twitter functions to create a file filled with tweets. For performing Markov analysis, I parsed through each line in the file of tweets and normalized all tweets such that any unwanted data would be removed, such as extraneous headers or urls, and made all characters lowercase. Additionally, I noted in the Markov analysis when a word was at the end of a tweet, so that in text generation my program would have a better understanding of how to end sentences. All of the data from the Markov analysis was saved into a dictionary of keys composed of the words found in the tweets, which mapped to dictionaries of the words found following the key words, mapping to the frequency of the words following the key words. The Markov dictionary was then saved and pickled to a text file.

To synthesize text from the Markov dictionary, my program would choose a word to begin a sentence with, and proceed to randomly choose the next word out of following words, with more weight applied to following words that had higher frequency. I had originally planned to create strings of words based purely upon the most frequent consecutive words (thus the modified selection sort), but later decided that having an element of randomness could lead to more interesting results. The program would then return a list of all the strings of words it had created, while also pickling and saving it to a text file.

Results:

In terms of the data analyzed, I initially only chose tweets that included the phrase “dankmeme”, as my intent of the project was to create a meme generator. While some of the strings created as a result of the Markov synthesizing made sense and were humorous, many lacked grammatical structure or included some profanity. The profanity can easily be filtered out, but in order to create better grammatical structure, I then chose to include data from other arbitrary Twitter search queries, but put more weight on the dankmeme data during text generation. Unfortunately, the results of adding more arbitrary Twitter data weren't particularly successful either, with many strings including words from other languages, and the grammatical structure was not noticeably improved.

Overall, many of the generated strings ended up as nonsensical strings, but they were still humorous in their content. For example: 'quand on keep going on. there you know how am i am a toilet.', 'one use the first thing i agree cause first lady president', 'memes keep popping up w/ ur right at the feels', and 'meme, we love political memes, rare pepes, trumps, and taken it is what'. There were certain strings created, however, that could one day be accepted as memes, such as 'mlg song', 'certain scent of those holiday deals', 'the hood prank gone wrong!', 'hate normie', 'bring a dankmeme train goin anywhere', 'im going guys. i really suck', and 'one must be in pepperoni'.

Reflection:

Overall, this project was successful in creating a simple version of a text synthesizer from a relatively small set of data. However, I could improve on the sentence structures generated by not only choosing the next word based on the frequency of words succeeding a given word, but by also comparing how often words are associated with each other in a given sentence. I also could generally improve my results by analyzing a larger set of data. Despite some successes, I believe my project's scope was too small, as I was only able to commit a couple of hours to create the project due to a very hectic week so I was unable to create better structured sentences and did not get a chance to implement my idea of pulling data from Twitch chat to auto-generate Twitch chat messages. The only unit tests I performed was on the Markov analysis of data I had read in, but I tested the function and modified the function until I was satisfied with its result. Going forward, I can use my new knowledge of Markov analysis to predict trends in data, or improve upon my version of Markov analysis to create more associations between data points.