Haley, Maire, and Mafalda

1. Project Overview: What were you trying to accomplish? What was your general approach?

Overall we attempted to create a program that would compare the overall sentiment of posts containing specific items such as a hashtag or a term, visualize this data, and have a graphical user interface. We began by creating a program that would search for tweets containing a specified term from twitter and then placed a certain number (which we would set) of them within a list. We then created a program that would filter out the tweets so that hashtags would be removed as well as random characters and words that were not in english. Then we ran a sentiment analyzer on the tweets and computed the average sentiment values for each set of tweets. We created a gui that a user could call this function from so that a person could choose which terms s/he searched for and how many tweets were going to be used to compute the average sentiment of the tweets. Lastly we used matplotlib to create a bar chart from the output of this function that would show (and compare) the average positivity and the objectivity of the two queries.

2. Implementation: How does your code work? What libraries did you use? How would someone (for instance a NINJA) run your code? What data structures (e.g. lists, dictionaries) did you use in your program and why?

When you run the main code, a gui pops ups prompting the user to input three terms. The first box is what the first term to search on twitter will be, the second box is the second term to search on twitter will be, and the third box is how many tweets will be used to find the average sentiment for each of the searches. Then a bar chart will pop up that will compare the average positivity rating and the average objectivity rating for each searched term.

Functions:

- get_tweets
    - Input the term you wish to search for and the number of tweets you want to search and this one outputs the tweets in a list
- filter_tweet
    - Input a “raw tweet” and outputs a filtered tweet
    - We remove mentions(@), tags(#), links, ‘RT’s, non-english words, and single letters excluding ‘a’ and ‘i’
    - We also make everything lowercase
    - We use a dictionary (py enchant english dictionary) to test whether words are english or not
    - This function outputs the tweet in plain english
- get_avg_sentiment
    - This one inputs a list of sentiments and it outputs the average sentiment
    - It removes trivial entries which are the ones that are [0,0]
- analyze_sentiment
    - inputs are things you want to search for and the number of tweets you want to search for and outputs the average sentiment
    - It calls the previous functions to do this
- make_graph
    - Inputs are the average sentiments and the terms searched for and it outputs a bar graph
- on_ok

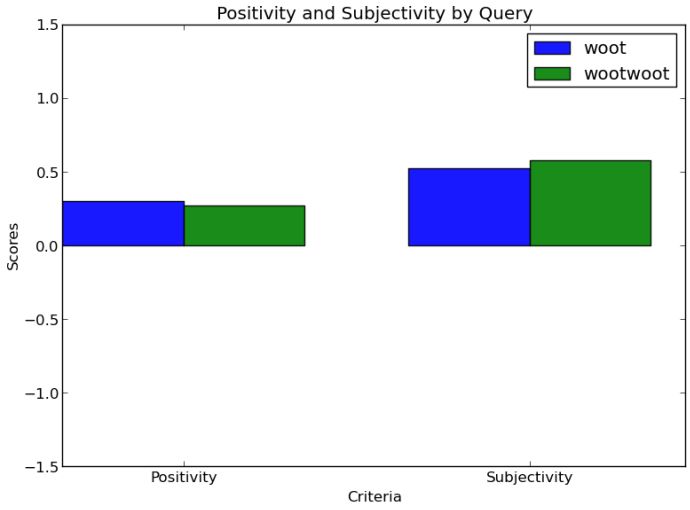
- This function creates a gui and tells the gui to run the previous code (ie search tweets and generate a graph) when the ok button is pressed

Data structures:

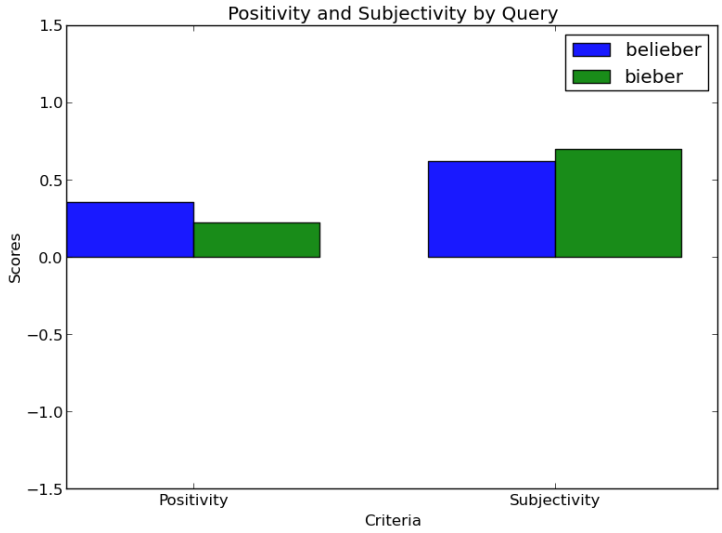
- Lists:We put the tweets we found in a list
- Strings: The individual tweets were put in strings that were placed within a list
- Dictionaries:Just one for the english dictionary to filter out word fragments, hashtags, etc.
- Tuples: its the output of the sentiment analysis and the input of what we were using in matplotlib
- Libraries: We used the libraries tkinter, matplotlib, pattern, and enchant.

3. Results: If you did some text analysis, what interesting things did you find? If you created a program that does something interesting (e.g. a Markov text synthesizer), provide a few interesting examples of the program's output.

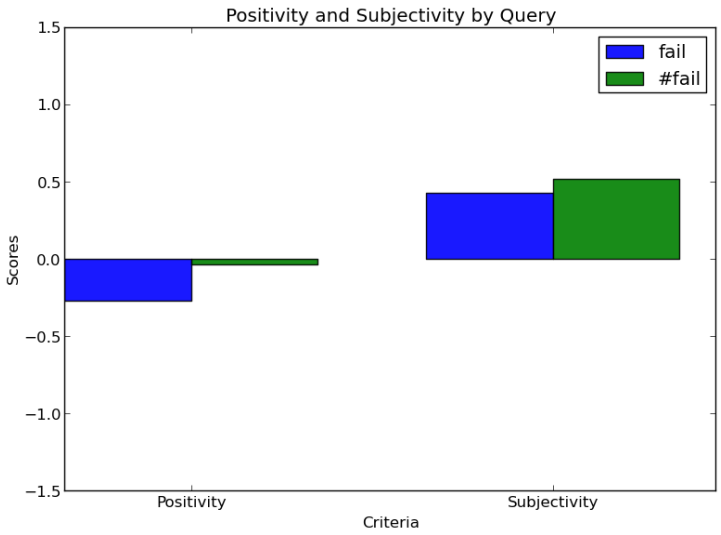
Woot is definitively more positively used than wootwoot on twitter.
[To Maire's shock and dismay. She will make a twitter account specifically to change this.]



Belieber and #Belieber are both persistently positive and very subjective.
Bieber and Belieber and are constantly positive.[Maire is not as sure she can challenge this successfully.]



Also fail was more negative than #fail in terms of positivity rating for the sentiment analyzer.



4. Reflection: from a process point of view, what went well? what could you improve? For instance, were there specific strategies for better coordinating on a software project with a partner? Was your project appropriately scoped? Did you have a good plan for unit testing?

What went well is we started early and we had a plan for what we wanted to accomplish so we could work on our own as well. Still we managed to have a long session on Sunday to finish the program because of scheduling conflicts.
We also had some issues with pushing to Git, so we ended up having a master-computer on which we kept all of our final code. We worked individually on sub-tasks or explorations on our other computers and showed results to one another during work-sessions, then emailed final/working code with appropriately meshing inputs and outputs to each other. In the future, we would really like to get Git working…
Our project, while relatively simple we think for more experienced programmers, took us a decent amount of time to figure out, but we ended up learning a lot, particularly about GUIs and methods/packages for data visualization, so we think it was decently well scoped.
We did not actually plan out our unit testing very well beforehand, We just tested along the way, printed the outputs, and then looked at the error messages. We tested each function individually and then when added it to the main script once it was working. On a larger scale, we also tested each large block of functionality (finding and filtering, the GUI, and the data visualization) individually with dummy data/inputs before hooking them up to one another.