CS 410 Project Documentation: Twitter Sentiment Analysis with Visualization

Team: Working Alone v2
Member/Captain: William Shih (ws21)

My free topic is about sentiment analysis of Twitter with data visualization. The Twitter API is used to extract Tweets from Twitter and perform sentiment analysis. I use JavaScript and D3 for the web page and data visualization. Python is used for sentiment analysis. Flask is used to send data from JavaScript to Python and from Python to JavaScript. I host the application on an online server. The application can be assessed at http://williamshih9.pythonanywhere.com/

VADER and TextBlob are pre-trained models used to perform the sentiment analysis. Both are available in Python. VADER (Valence Aware Dictionary and sEntiment Reasoner) and TextBlob are lexicon and rule-based analysis tools that can measure sentiment. For every tweet, I obtainsix scores from VADER and TextBlob.

Four scores are provided for VADER: compound, negative, neutral, and positive scores. The VADER compound score is the overall sentiment. Two scores are provided for TextBlob: polarity and subjectivity. Subjectivity measures how fact-based or opinion-based text is and TextBlob polarity is the overall sentiment.

I allow the user to pick 4 queries of Twitter to allow the user to compare the sentiment of the different queries. Two types of queries are allowed. The user is allowed to search Tweets by filtering for a single hashtag or a single string of text. This type of query allows to grab tweets from up 7 days ago from all Tweets in English. The other type of query is by Twitter username, which search tweets from a Twitter username. For simplicity, 1000 tweets are obtained for every query.

To avoid the requirement of having the Twitter API must pull tweets every time, I pre-compute Twitter sentiment data for a variety of users and queries (71 queries in total). This is useful since the Twitter API is limited to 500,000 tweets per month and it would be a problem to have to use the Twitter API every time to do a comparison. I also provide sentiment data for a large sample of 30,000 Tweets that I call the “Baseline” for comparing queries with the baseline.

Code

The source code for the front-end application is in "Project Application/templates/index.html" and the source code for the back-end portion is in "Project Application/app.py". The pre-computed queries can be seen in the “Project Application/data_sep” folder. The application can be assessed at http://williamshih9.pythonanywhere.com/

How the program works:

1. User either types a query or selects one of the pre-computed queries. If a query is pre-computed, then then only the pre-computed query will be used, so more recent Tweets (past November 25) will be not used.
2. The user must select 4 queries. No less and no more to keep the program simple. If any of the 4 queries does not return matching Tweets or is not a valid request, then the visualization will be not displayed, and the user will enter the 4 queries again.
3. If the 4 queries are successful, 6 stacked bar graphs will be displayed for each of the 6 scores from VADER and TextBlob. Each stacked bar graph contains 4 stacked bars (one for each query). Each stacked bar contains the distribution of the scores for that query. For “polarity” and “sentiment”, there are 20 bins (one for each 0.1 bin from -1.0 to 1.0). For the other four, there are 10 bins (one for each 0.1 bin from 0.0 to 1.0). Each stacked bar graph will be on its own separate section and the user can toggle between each of the graphs easily.
4. The user can hover over the bars to see more information (tooltip). 5 pieces of data are displayed: 1) Percent of Tweets inside that score bin. 2) Percent of Tweets below that score bin. 3) Percent of Tweets above that score bin. 4) Mean score of the query 5) Average number of characters per tweet of the query. This data allow the user to compare the distribution of scores between the 4 queries easily.
5. On the bottom of the page, the number of Tweets per query and also the timestamp of Tweets (start time to end time) per query is also shown.

Evaluation

The program works as expected and for the most part, all goals were achieved with this project. The website works as expected and the analysis given from VADER and TextBlob seem reasonable given a large sample size of Tweets. Since VADER and TextBlob work quite well for analyzing social media already, it did not feel appropriate to train my own model that perform worse than those two models. There is simply not enough training data or computing resources to build a good model myself.

Contribution of Each Member

The project took about 30 hours to complete, which is what the workload was expected from the Project Proposal and well above the 20 hours requirement per person in the Project Instructions.