Sentiment Analysis of Facebook Posts

Patrick Huston - February 26, 2015

Project Overview

Utilizing the pattern.web API, I accessed Facebook user and post information to gain insight into correlations between user demographics and average post sentiment.

Implementation

Several key components comprise the architecture of my implementation:

The first step in accessing data from the pattern.web Facebook API is to create a facebook session, and from there, several different objects can be accessed. The two I am interested in are FRIENDS and NEWS, and in order to create some value from these two elements, some sort of data structure must be created to combine the two. After accessing the ids of all friends, their user profile objects can be accessed as well, in the form of tuples. Given the fact that the ultimate goal was to find correlations between user's demographic information, and their post information, I created a dictionary with keys as friend's user profile objects

The next step was to gather the relevant information for each user profile object - the total number of posts, the average sentimentality, and the average subjectivity of each post. This was achieved by parsing through every NEWS object that was returned from a search for NEWS with the user's id. To avoid diluting the sentimentality results, I created a filter that discards posts that fall into seven different categories:

1. Facebook auto-generated updated cover photo posts
2. Facebook auto-generated life event posts
3. Facebook auto-generated shared photo posts
4. Facebook auto-generated shared video posts
5. Facebook auto-generated shared link posts
6. Facebook auto-generated comment posts
7. Posts that are entirely URLs

This filter was implemented using regular expressions, which successfully created a subset of posts that do not include meaningless auto-generated posts made by Facebook. Finally, with this dictionary, I tested several different comparisons between user demographics and post analysis. Ultimately, due to the lack of publicly available birth dates and years, the only feature that was available often enough to produce a statistically significant result was user gender.
Next, I analyzed average positivity of posts over the 365 days in a year. Instead of successively making calls to the Facebook API every time I wanted to analyze results, I implemented a MySQL database to locally store API calls for further manipulation and analysis later. This database is comprised of just

one table that maps each post's day of the year to it's positivity. Later, this table is read and parsed in Python, and a dictionary is created that represents each day, and the corresponding average positivity of all posts for that day. Finally, I plotted these results using the matplotlib library.

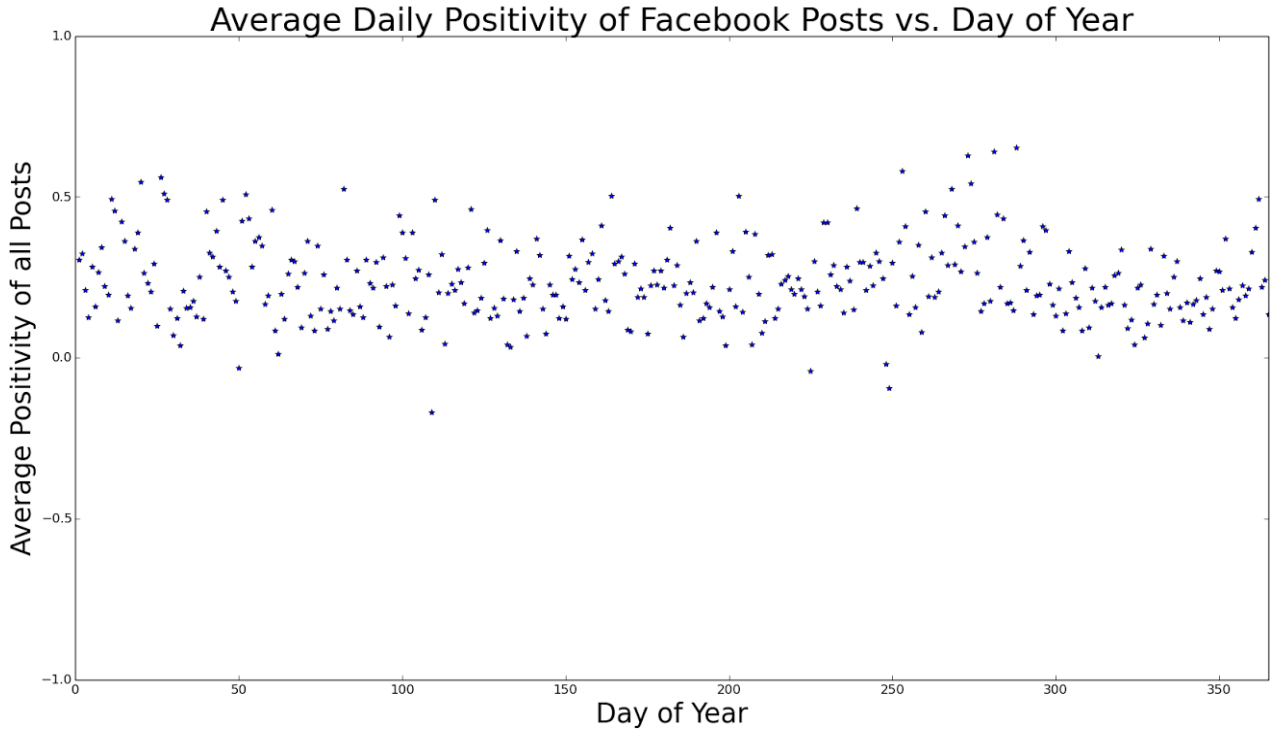
Results

After gathering data on friend's posts, the following results were gathered. Compiling all posts for female and male friends, the average positivity and subjectivity for each gender can be calculated:

Average male positivity: 0.202661588253
Average female positivity: 0.278054690458
Average male subjectivity: 0.371246516527
Average female subjectivity: 0.427963103535

While these results are not groundbreaking or particularly interesting, it does make sense based on my observations looking at Facebook posts. On the whole, my female friends tend to express more positive, complimentary posts directed at friends. For example, posts akin to "Stop it, you're so beautiful" are common among my female friends, and not as much for male friends.

The following graph represents the second analysis of posts I conducted. For this analysis, I created a dictionary mapping each day of the year to the average positivity of posts for that day:



Evidently, there is little correlation between the day of the year and the positivity of posts. The values tend to fluctuate wildly, and while it appears there may be a small downward trend near the end of the year, it is not nearly significant enough to base any real claims on. This could be due to several different factors. First, the sentiment analysis implementation provided by the pattern API is not great - parsing through individual posts, and comparing my understanding of them versus pattern's analysis, I found that the accuracy was not great. Additionally, many of my facebook friends reside in California, which decreases the likelihood of observing decreased positivity in winter months due to SAD (seasonal affective disorder). In conclusion, according to my analysis, my facebook friends on the whole don't tend to vent their sadness on the whole through social media.

Reflection

Reflecting on how this project went for me, I have several insights on the positive elements of my approach, and some of the elements that could be improved. First, I believe I did a good job of spending time prior to implementation brainstorming different possibilities of implementation, including thinking about what data structures would be appropriate for the different possibilities. Additionally, I did a reasonable amount of unit testing, especially as I incrementally developed different elements of my implementation. One aspect of my implementation that I am particularly pleased with is my implementation of a MySQL to locally store API call results to speed up analysis of this data. For future projects, I would like to spend more time making my code more readable, especially as I write it. The difficulty of commenting code increases dramatically as time passes from initially writing it.