Sentimentality of Inaugural Speech
Software Design Project 3 Text Mining

Ben Ziemann

February 23, 2017

1 Project Overview

Using Yale's Avalon resource, I was able to obtain the inaugural speeches of all the presidents. Using a combination of BeautifulSoup to get the links and html, various string parsing techniques to read the speeches, and Vader Sentiment Analyzer, I was able to find the positivity and negativity of each speech. I hoped to compare these sentiment values to one another to see if there was any correlation between these values and the president's popularity.

2 Implementation

The first step was somehow obtaining the inaugural speeches from a common resource. I found "The Avalon Project" from Yale as a listing to all the speeches. However upon looking at the HTML I found that the links were split up, with the base being a Yale Avalon string and the particular speech link in a different part. Using BeautifulSoup and Python Regular Expressions, I could get the HTML from the page, looking for unknown strings (the particular links) between known HTML tags. I saved these in a list, from which I iterated through and created a text file for each of the found links, with a sliced version of the speech in it. I now had each speech in a separate file on my computer in plain text. Once this process has been run once it will not repeat due to a check for these files' existence at the beginning. This prevents the program from requesting the speeches too many times.

Next was a matter of reading each of the files and running it through the sentiment analyzer. Iterating through each file in the directory I saved the speech text files to, I could process them one at a time. The sentiment analyzer returns the sentiment values as a dictionary, with the type of sentiment being the key and the value being the value. Using this I targeted the specific keys and values I wanted and combined with with the president's name from the file name to create an easily readable output for each president's speech. Because this output is in the form of a list, it is easy to work with, allowing for sorting and access of the data in it.

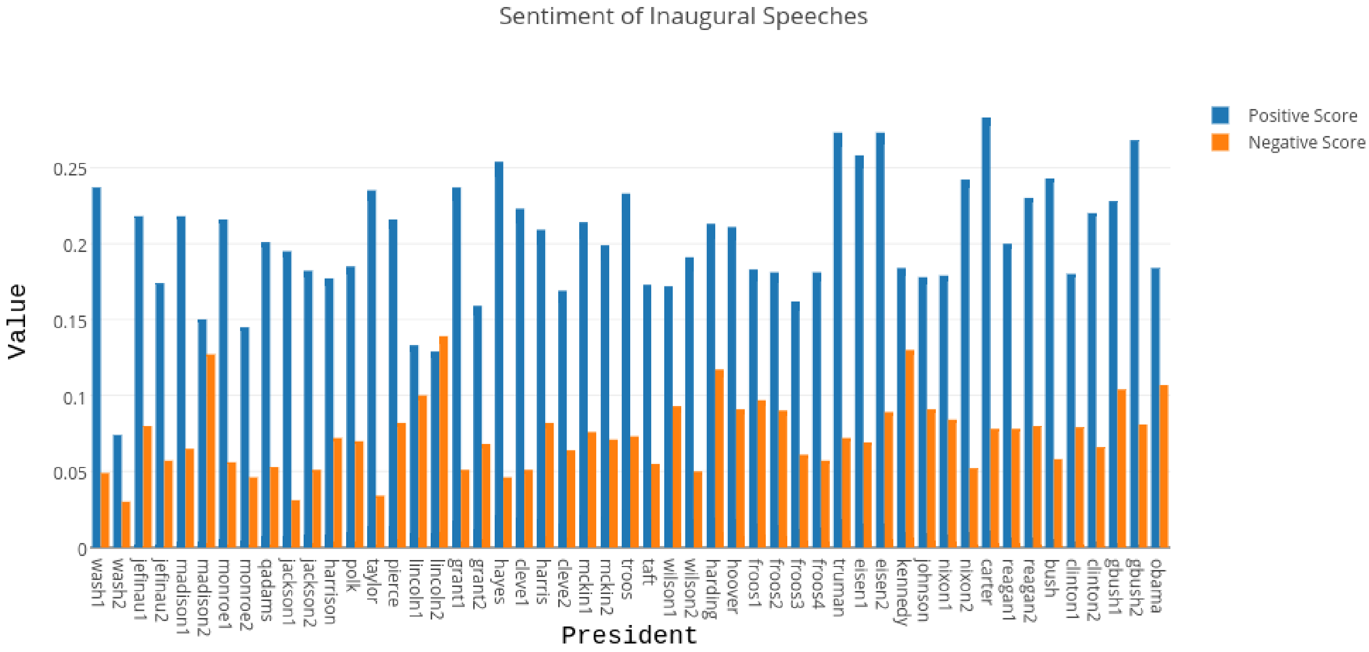
Finally I used plotly to make an easy to read bar graph of both the positive and negative scores of each speech (see below).

One of the choices I made during the design was the decision to iterate through a series of text files storing each of the speeches, as opposed to a single file holding them all. This allowed me to much more easily manipulate the order of the speeches and the speeches themselves. It also allows much easier access to specific speeches, being able to immediately open the desired speech, instead of parsing through one giant document.

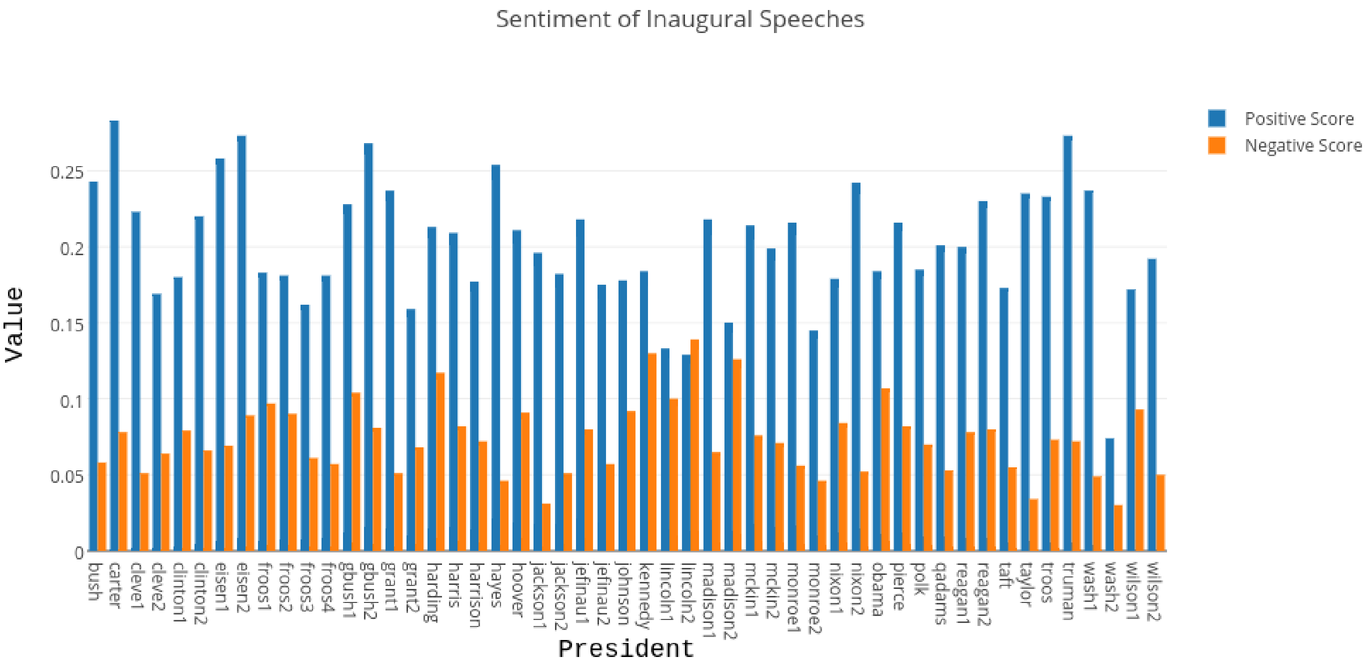
3 Results

```
(’adams’, ’Positive: 0.0’, ’Negative: 0.0’)
(’buchanan’, ’Positive: 0.0’, ’Negative: 0.0’)
(’bush’, ’Positive: 0.243’, ’Negative: 0.058’)
(’carter’, ’Positive: 0.283’, ’Negative: 0.078’)
(’cleve1’, ’Positive: 0.223’, ’Negative: 0.051’)
(’cleve2’, ’Positive: 0.169’, ’Negative: 0.064’)
(’clinton1’, ’Positive: 0.18’, ’Negative: 0.079’)
(’clinton2’, ’Positive: 0.22’, ’Negative: 0.066’)
```

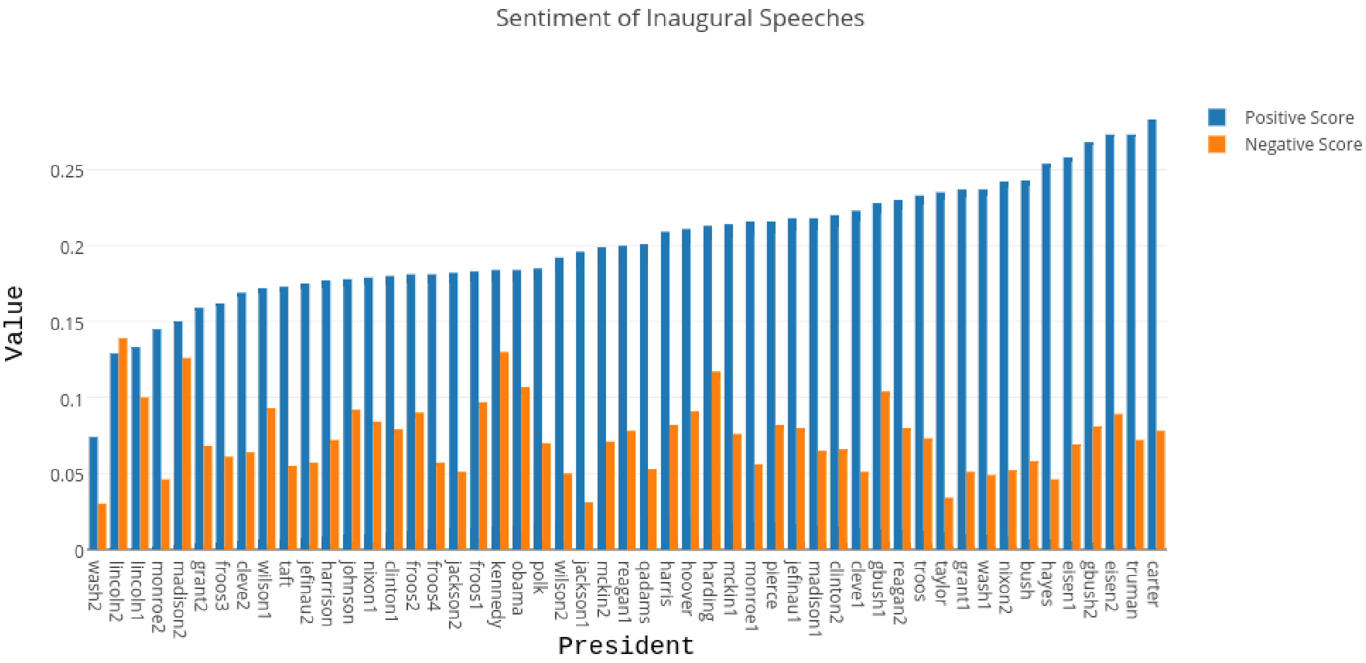
Above is a small sample of the output of the program in text format. When graphed using Plotly to visualize these scores, the data becomes a lot more easier to compare:



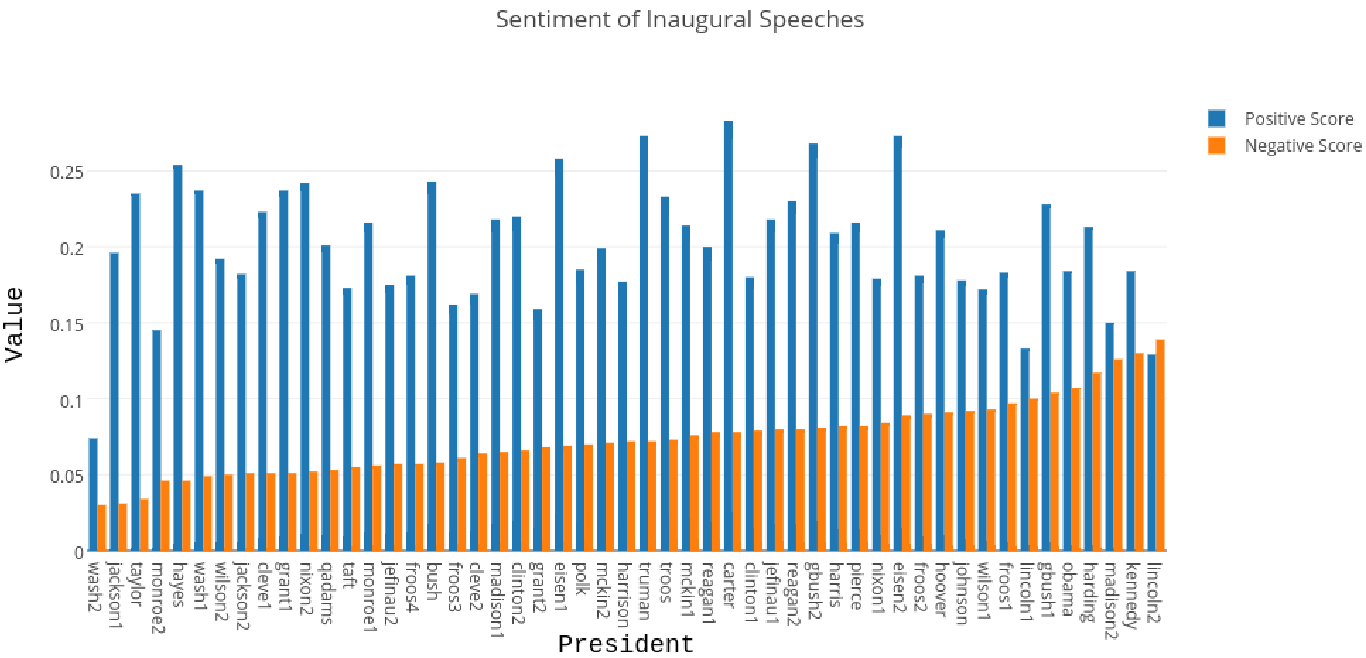
Bar graph of the scores in chronological order



Bar graph of the scores in alphabetical order



Bar graph of the scores in order of positivity



Bar graph of the scores in order of negativity

Above are the scores of each speech sorted by alphabetic order of the president, positivity, and negativity. Note that the speeches without data were a result of the previous president dying in office, so the president did not give an inaugural speech. As expected, the greater the positivity of the a given speech, the less negativity there was. More interesting though is that the sum of the positive and negative scores for each speech across all speeches are not equal, leading to the idea that some speech held more feeling than others.

4 Reflection

Overall I feel I accomplished my goal and got solid results back in the form I expected to. Once I got the texts they were easy to format for use in the sentiment analyzer. The output of the analyzer was also easy to manipulate so that I could create the graph. All of this became much easier when I took it step by step by defining new functions. By far the most difficult part of the project for me was creating the crawler to find the links on the Avalon site.Most of my time on this project was looking through a few of the recommended package's APIs to find functions that did what I wanted, and then learning how to use them, specifically the Python Regular Expressions. It definitely would have been nice to know what resources were out there in terms of functions but now that I do know I feel I can do so much more in future projects. To improve upon this project I feel I could better format the graphs for readability. Additionally it would be interesting to see if there was a general trend in positivity or negativity chronologically.

Overall I am very pleased with what I have created and learned a lot about integrating multiple packages into a working code as well as breaking down the project into simpler steps.