Election 2016 Twitter Sentiment Map

Alex Engel
Stanford University

ABSTRACT
Many political polls conducted today are very prone to selection bias and other factors that negatively impact the validity of the poll results. Because of this, we need to be constantly searching for new, more valid ways to capture the collective opinions of the nation. In this paper, I will discuss the development of a political sentiment visualization that is generated using data collected from Twitter. This visualization utilizes both historical and real-time data in order to create a portrait of the nation's sentiment concerning the three remaining frontrunners in the 2016 presidential election. By creating this visualization, I hope to give a more unbiased view of how the population of the United States regards the remaining candidates in the 2016 election cycle.

Author Keywords
Sentiment, map, data, visualization, Twitter, real-time, historical, tweet, election

INTRODUCTION

Today, most political polls are conducted either through surveys or brief interviews. Unfortunately, both of these methods are very prone to selection bias, leading questions, and other factors that affect the validity of the poll. Consequently, we should constantly be on the lookout for new methodologies that allow for the creation of more unbiased views. Another downside to traditional polls is that the results are often displayed as raw data. Although information-dense, this raw data is often difficult to draw conclusions from due to the absence of an effective way to browse and compare different aspects of the data. Additionally, the presence of raw data allows people to pick and choose the data that analyze or share. Depending on the way that this data is picked and chosen, the resulting conclusions drawn can be highly variable. Because of this, an effort must be made to provide some sort of polling methodology that can provide more unbiased results.

In an attempt to solve the problem of bias, work discussed will attempt to capture the nation's political sentiment through a different medium, social media. When one posts on social media, they are not doing so with the intention of providing data for polling purposes, and thus it can be postulated that data captured from these sources will have a lower instance of methodologically induced bias. However, social media data will not be completely free of bias. For example, there is still inherent selection bias due to the fact that the population active on social media is not representative of the world's population as a whole. That being said, it is still likely that data collected from social media has the potential to be closer to the unbiased ideal than data collected competing methods of polling.

The work discussed in this paper will take the form of map that conveys information about the nation's sentiment regarding one of the three remaining candidates in the 2016 presidential election, Donald Trump, Hillary Clinton, or Bernie Sanders. To do this, this web application will use data collected from the popular social networking site, Twitter. On Twitter, users post short, directed strings of text called tweets for their followers and the general public to read. The tweets collected for this project will all be regarding the three remaining frontrunners. As each tweet is collected, its text will be analyzed to see if any conclusions can be made about the binary sentiment of the tweet. In other words, the overall attitude of the tweet will be classified as positive or negative.

Once this data is collected, it must be visualized in a way that allows the viewer to see national trends at-a-glance, while still allowing them to interact with and examine the more low-level aspects of the data. In order to do this, the sentiment map will show both historical and real-time data. Historical sentiment data will be grouped by location and averaged by state. Then, this data will be encoded into a color scheme that will fill in each state with the color regarding the overall sentiment. Real-time sentiment data will be plotted as it is collected. Each incoming tweet will be analyzed for sentiment and then immediately plotted on the map through the use of location metadata included in the tweet. These data points will use a binary color-encoding scheme to represent the sentiment of the tweet. The user will be able to interact with this data by choosing which candidate the historical and real-time sentiment data displayed on the map is related to. In addition, users can hover over states to see the averaged sentiment value in that area and can hover over real-time points to see detailed information about each specific real-time tweet.

Motivations

The main motivation behind the creation of this visualization is to show that technology can provide us with a means to draw less biased conclusions about a large variety of topics. Using a technology-based methodology to scrape already existing data has the potential to give us more unbiased results than other methodologies such as surveys that methodologically require the creation of new data.

Another motivation behind this project is to show that location-based sentiment data can be a very useful tool in a wide variety of situations. With just a few small tweaks, the presented visualization could be made to analyze the sentiment regarding a product, service, or idea instead of only showing political sentiment as it does now. This sentiment data could be very valuable to marketers, business analysts, and other decision-makers as they decide how to tailor their strategy to suit the sentiment of different areas.

The final motivation for the creation of this visualization is that its creation will require the use of a wide variety of technologies. Getting all of these technologies working consistently in unison is a challenge that will make the development of this visualization both interesting and difficult.

RELATED WORK

There has been a wide variety of work done in both analyzing the sentiment of tweets and plotting information about tweets on a map view. However, I have not come across any location-based Twitter sentiment visualizations regarding the 2016 election. This section will address some of the most relevant examples of previous work and explain how the visualization created for this project is novel.

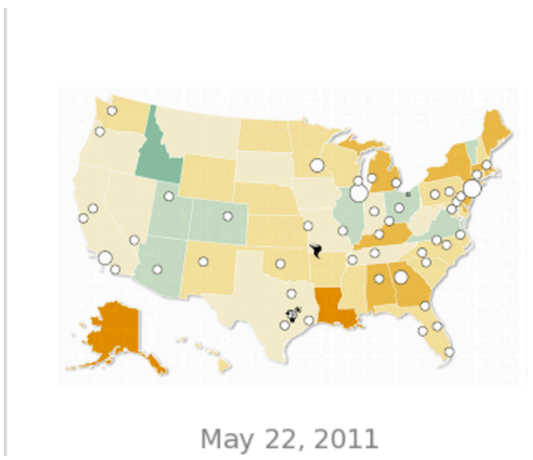


Figure 1

The first example of related work is a very close analogue to my project. It is a sentiment map regarding weather that was created by the Dialogue Earth project. As can be seen above in Figure 1, this map displays weather sentiment for each state as color shading. Additionally, it highlights areas where there is a large volume of weather related tweets being posted. Although this visualization seems like a very cool concept, it unfortunately was not operational during the development of my project and so I do not know many details about the interactive aspect of the visualization. My project will be novel from this visualization in that it will be in regards to the 2016 election instead of weather. In addition, it will allow for interaction with tweets at a lower level than this weather visualization allows for. Instead of highlighting clusters of tweets in an area, my visualization will allow for interaction with individual tweets that arrive to the map in real time.

The next attempt at sentiment visualization is quite different from what I will be doing, but definitely deserves a spot in the related works section. Created at NC State, the "sentiment viz" application allows users to search for specific term and stream tweets related to this term in real-time. As these tweets arrive, they are plotted in 2-D space where the x and y coordinates determine the sentiment of the tweet. The key to this application is that it uses a non-binary sentiment classification system, allowing for tweet sentiment to be classified as 'elated' or 'stressed. The visualization discussed in this paper will be similar in that it will have a real-time stream of tweets arriving for analysis, but will differ in the way that it plots the sentiment, using a map view instead of 2-D space. This visualization can be seen in Figure 2.

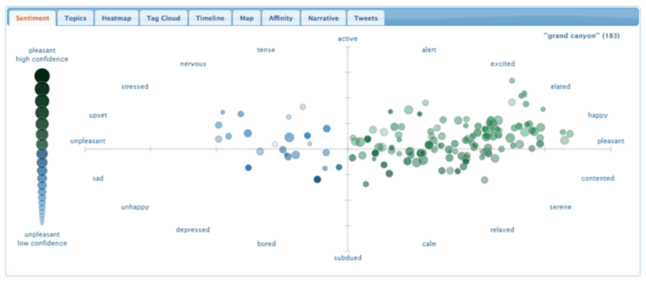


Figure 2

The final piece of related work that I will discuss is the OneMillionTweetMap. This visualization allows the user to input a search term and watch tweets related to this search term appear in real-time on a map view. Additionally, this visualization supports time filters and allows for inspection of each individual tweet. The way that my visualization will differ from the OneMillionTweetMap is that it will be focused on sentiment. The OneMillionTweetMap has no sentiment analysis aspect to its operation. This piece of work can be seen in figure 3.

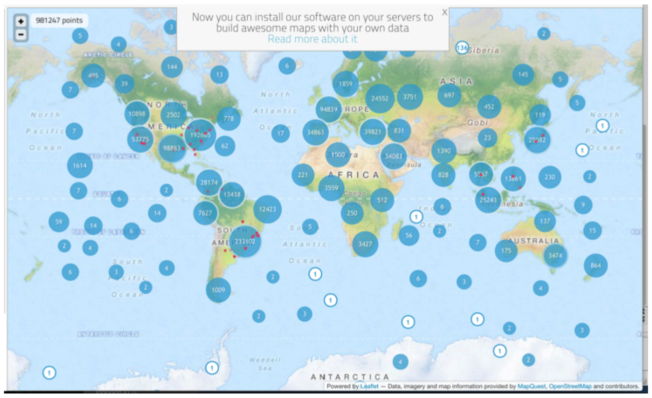
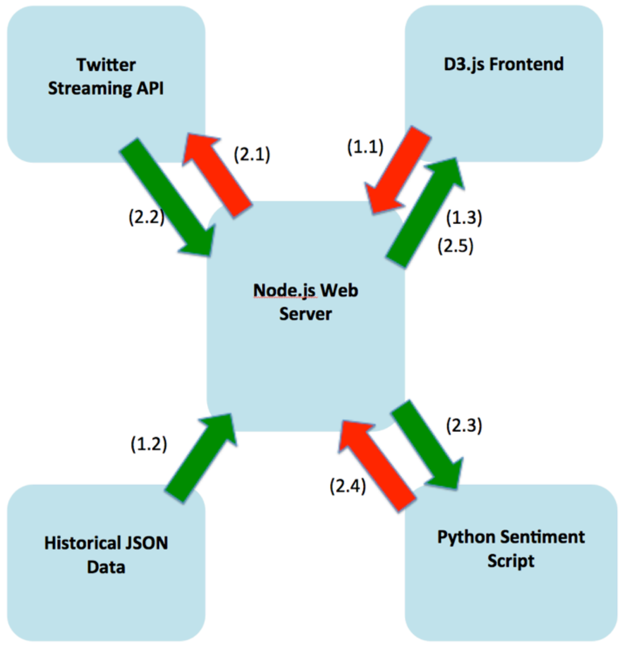


Figure 3

All of the related works described above have some aspect that will be integrated into my visualization. However, none of them will be able to actually replace my project, and thus the generated visualization will be novel.

METHODS

As mentioned earlier, this visualization takes the form of a color-coded United States map that is populated using Twitter data. There are 5 main components to this web app. The overall system architecture is show in the visualization below and is accompanied by a list denoting the task flow for visualizing both historical and real-time data.



Task 1: Historical Sentiment Data

(1.1) D3 frontend sends request to server with selected candidate

(1.2) Server analyzes historical JSON data

(1.3) Historical sentiment data returned to server to color states

Task 2: Real-time Sentiment Data

(2.1) Server requests relevant tweets from Streaming API

(2.2) Server receives and processes tweets

(2.3) Python script analyzes each tweet for sentiment

(2.4) Sentiment returned to server

(2.5) Tweet and sentiment sent to frontend via Web Sockets for plotting

Twitter Streaming API

The Twitter Streaming API is a very powerful tool for data analytics. To start the stream, a consistent TCP/IP connection is first made between the application server and Twitter's streaming service. When creating this connection, the application server also passes a set of query strings to the streaming service that dictates which tweets will be returned. In the case of this visualization, query strings were set so that only tweets containing the name or Twitter handle of the relevant candidate(s) were returned. However, there are several limitations to the Streaming API. First of all, less than 10% of all tweets are streamed to clients of the API. This is due to the fact that streaming all tweets would take an enormous amount of bandwidth and additionally would prevent Twitter from charging for use of their Firehose, a contemporary of the Streaming API that does allow clients to stream ALL public tweets in real-time. That being said, the small portion of tweets being returned to the application server is still a large enough quantity to allow for robust results. In one week of gathering data, about 15,000 tweets were collected. Another limitation to the Streaming API is that clients can only supply one type of query. This was a significant issue that arose when designing this visualization as it prevents one from querying both by tweet text and by location. This limitation necessitated several drastic changes in the proposed visualization design as the project progressed.

Historical JSON Data

As stated above, this web app uses both historical and real-time data to create an effective visualization. The key to presenting this historical data is a JSON file containing a list of historical tweets about the selected politicians. This JSON file also contains the associated sentiment value for each tweet. This data was collected from May 25-June 3, 2016 using the Twitter Streaming API. As tweets arrived at the web server, they were analyzed for sentiment and then placed into a MongoDB collection. To allow for ease of access to the data, this MongoDB collection was then exported to JSON format so that the overall sentiment for each state can be analyzed by the web server and sent back to the frontend.

D3.js Frontend

The user-facing frontend of this web application was created using the D3.js framework. D3 (Data-Driven Documents) is especially well geared to this undertaking due to its ability to quickly and easily link many types of data to the document-object model (DOM). In this visualization, D3 has three main tasks. First, it handles the drawing of the map using scalable vector graphics and GeoJSON paths. Next, using data retrieved from the Node.js web server described below, D3 handles the shading of the individual states on the map. To do this, the average sentiment of all tweets from each state is computed. After this computation is finished, D3 makes use of a linear color scale in order to convert the numeric sentiment value to a corresponding color value that is then used to fill the state. The third job that D3 handles in this visualization is the plotting of real-time tweets. As tweets arrive to the frontend (via Web Sockets), D3 extracts the location data from the tweet and uses it to plot a point on the projected map. Additionally, D3 determines the coloring of the newly plotted point based on the sentiment of the incoming tweet. This time, instead of using a linear scale to color the points, we use a binary scale that simply classifies tweets with positive sentiment as 'green' and tweets with negative sentiment as 'red'.

Python Sentiment Analysis Script

The key to this entire visualization is a Python script that runs a sentiment analysis routine on every incoming tweet from the Streaming API. In order to do this sentiment analysis, the Python script uses a natural language processing library called TextBlob. This library uses machine-learning principles to analyze the words in a tweet and decide whether or not the overall message is positive or negative. This sentiment is outputted on a scale from -1 to 1, with -1 correlating with negative sentiment, 1 with positive sentiment, and 0 with neutral sentiment. Upon finishing the analysis, the resulting sentiment is sent back to the web server to be either stored in the MongoDB collection or sent to the frontend for plotting. One of the limitations of this Python script is that the TextBlob sentiment analysis function is based on a corpus of words pulled from a set of movie reviews. Obviously, the words in movie reviews will be drastically different from most words in political tweets. Because of this, the sentiment value returned by the Python script will not always be accurate. That being said, when examining the output of this script alongside the given tweet, I found that the accuracy of this analysis was most definitely better than a random classification and thus, with a large enough dataset, the overall sentiment for each state should be quite accurate.

Node.js Web Server

At the core of this visualization lies a Node.js web server. This web server interacts with all 4 of the other visualization components described above. The server handles three main workflows. First, when the frontend is loading, a request for historical data is sent to the server containing a parameter that tells the server which candidate is currently selected on the frontend. Using the routing framework provided by Express.js, the server captures this request and reads a JSON file containing tweets about the selected candidate into a Javascript object. Once this file is read, the server groups tweets by state and finds the average of sentiment value of the tweets from each state. After this analysis is done, a dictionary containing each state and it's relative tweet sentiment is sent back to the frontend to be converted into color values used to fill in the states.

The rest of the server is devoted to dealing with real-time data. To begin the streaming process, the user first must sign into Twitter on the frontend. Once signed in, the server retrieves the user's access key and access key secret and uses this information to initialize a new Twit instance. Twit is a Node.js framework used to connect to the Twitter REST and Streaming APIs. Once this instance is established, the user can select which candidate(s) they would like to stream tweets for. After a selection is made, the frontend sends a message to the server with the selection via Web Sockets and the Socket.io framework. When the server receives this socket message, Twit sends a new request to the Streaming API for tweets regarding the selected candidate. As each tweet is pushed to the server, a child process is called to launch the Python script used for sentiment analysis. Once this analysis is complete, the tweet information and sentiment information are sent back to the frontend, again using sockets.

RESULTS

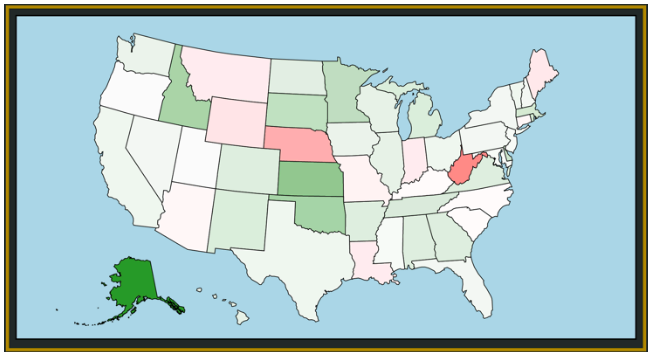
Overall, I was very pleased with the end product of my visualization. In addition to being completely functional, it shows national trends clearly and the use of the real-time feature has shown some interesting relationships between political events and tweet sentiment. I will group my results into 3 categories.

Functionality

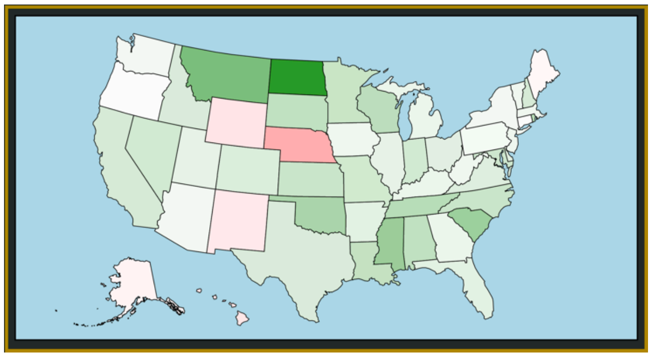
As I am writing this paper, I am happy to say that the visualization functions exactly in the way that I intended it to. Upon entering the application, the user is presented with a fully colored map view that defaults to display overall state sentiment information for one of the candidates. The user can interact with this visualization in two ways. First of all, they can change a radio button that determines which candidate the state sentiment is being displayed for. Next, the user can choose to display real-time tweets regarding one, two, or all three of the potential candidates. They do this by selecting checkboxes for the candidates that they would like to follow in real-time. Upon doing this, data points start appearing on the map and allow the user to hover over them to see information directly related to a singular tweet.

Effectiveness of Showing Trends

Sentiment analysis is still in its infancy, and as such, has a ways to go in terms of accuracy. That being said, by drawing from large amounts of data and plotting it effectively, my visualization shows definite trends in how political sentiment about the remaining candidates is distributed throughout our nation. One of the more obvious trends is the ambiguity of Twitter's sentiment regarding Hillary Clinton. As can be seen in Figure 4 below, tweets about Clinton have high sentiment variability between states. This is in stark contrast to the map that results from selecting Bernie Sanders, which is much more uniformly colored (see Figure 5).



Clinton (Figure 4)



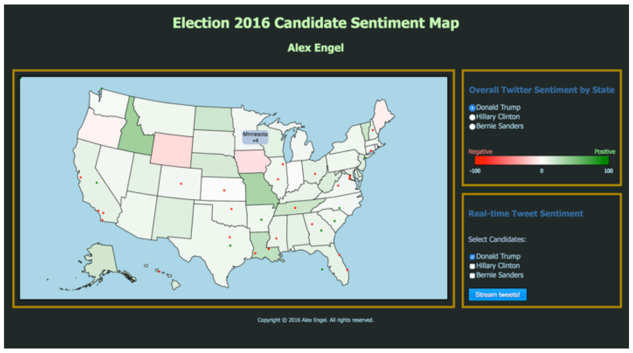
Sanders (Figure 5)

Although such trends can definitely be seen thanks to my visualization, I was hoping that there would be more visible trends in different regions of the country. For example, I would have expected the more liberal west coast to show a positive tendency towards democratic candidates, while the Midwest and South would should a more negative overall sentiment about these candidates. Unfortunately, such regional trends were hard to pinpoint.

Real-time Results

One of the most interesting results of my project was due to the real-time sentiment feedback that is plotted on the map. During development, I noticed that there seem to be an overabundance of tweets Bay Area that had positive sentiment about Bernie Sanders. After observing this for a while, I realized that Sanders was holding a rally in Oakland during this time, as most of the tweets that could be seen on the map were regarding a rally. Consequently, one of the unexpected results of my visualization is that you can pinpoint where certain political events are happening in real-time. During development, I thought that the fact that I was only processing ~10% of tweets would prohibit my visualization from having this type of resolution in regards to real-time data.

A screenshot of the final visualization can be seen below in figure 6.



Election 2016 Candidate Sentiment Map

Alex Engel

Overall Twitter Sentiment by State

[✓ Donald Trump]
Hillary Clinton
Bernie Sanders

Real-time Tweet Sentiment

Select Candidates:

[✓ Donald Trump]
Hillary Clinton
Bernie Sanders

Copyright © 2016 Alex Engel. All rights reserved.

Figure 6

DISCUSSION

In relating my project back to the world of visualization as a whole, I think it is safe to say that the biggest takeaway is the notion that location-based sentiment visualization is relevant both at a macroscopic and a microscopic level. Through the macroscopic lens, location-based sentiment can show many interesting trends not just in regards to things such as elections, but also in regards to products and services. I hypothesize that as sentiment analysis becomes more and more accurate, it will become much more widely used in all fields thanks to the fact that macroscopic sentiment fluctuations can be seen with a single glance at a location-based sentiment visualization.

At a microscopic level, sentiment visualization allows us to see changing sentiment as it occurs. I mentioned the example of being able to pinpoint a Bernie Sanders rally due to the large influx of positive tweets about him in a certain area. This could be helpful for companies that need to alter their strategy on a day-by-day, case-by-case basis. By being able to pinpoint an area where there are extreme cases of growing positive or negative sentiment, users of real-time sentiment visualizations will be able to alter strategies on-the-fly in order to account for this changing sentiment.

Another large takeaway from this project is that sentiment analysis visualizations can take on a very large variety of forms. As explained in the related work section above, there are many types of sentiment analysis visualizations out there and all of them are geared towards a slightly different purpose. Obviously, my visualization is aimed at identifying location based sentiment trends, but others, such as the NC State visualization, are better for getting a more detailed description of the sentiment itself, instead of just a positive or negative value. Still other visualizations might show sentiment over time. Each type of sentiment visualization has its own strengths and weaknesses.

FUTURE WORK

Even with the perceived success of my finished visualization, there is still a lot of room for improvements in the domain of location-based sentiment visualizations. The first area for improvement that I would like to discuss is the potential for a more robust version of my visualization. By this, I mean a visualization that allows users to inspect the sentiment of not just tweets regarding the election, but of tweets regarding any subject. If such a tool were to be created, I am sure that many would quickly adopt it due to the fact that this type of visualization would not need any alterations to make it useful to a very wide variety of people. There are however, some obstacles that would make this visualization difficult to realize. The main roadblock is that in order to display historical sentiment data in any reasonable amount of time, there would be an absolutely enormous archive of tweets that the application has to store due to the fact that it has to account for any search term. A more realistic approach would be to make a robust visualization that only deals with real-time sentiment data.

Another area for improvement is in regards to the visualization itself. Right now, it serves its purpose very well, but I can envision a whole host of other filters that could be applied to tweets in order to allow closer inspection of sentiment detail. For example, a time filter would be a very useful feature to add, as would an additional view that shows the top trends among the tweets that are being analyzed.

The final area for improvement, although not dealing with visualization itself, still must be accounted for. As mentioned earlier, sentiment analysis is still in its early stages and consequently leads to varying degrees of accuracy depending on what application it is being applied towards. In the case of my visualization, the sentiment analysis script was trained on a corpus of movie reviews. Because there is likely a large disparity between words that appear in movie reviews and words that appear in political tweets, it is likely that a large portion of the tweets collected were incorrectly classified. Future iterations of this project could create a training set of politically based words in order to achieve more accurate results.

ACKNOWLEDGEMENTS

Thanks to Maneesh Agrawala and the rest of the Spring 2016 CS448B teaching staff sharing your knowledge of data visualization throughout the course of this project. Additionally, thanks to my peers in the course that provided helpful feedback during the progress presentation.

REFERENCES

1. Brewer, C. A. 1999. Color Use Guidelines for Data Representation, Proceedings of the Section on Statistical Graphics, American Statistical Association, Alexandria VA. pp. 55-60.

2. D3.js - Data-Driven Documents.

3. E. R. Tufte. Envisioning information. Optometry & Vision Science, 68(4):322–324, 1991.

4. Tufte, Edward R. The Visual Display of Quantitative Information. Cheshire, Conn. (Box 430, Cheshire 06410): Graphics, 1983. Print.

5. Express.js – Web Development Framework

6. Nicholas Kong, Maneesh Agrawala. Graphical Overlays: Using Layered Elements to Aid Chart Reading InfoVis 2012, October 2012. pp. 2631-2638.

7. MongoDB – NoSQL Database

8. Node.js – Javascript Runtime

9. One MillionTweet Map - http://onemilliontweetmap.com

10.https://www.csc.ncsu.edu/faculty/healey/tweet_viz/tweet_app/