Using Twitter to Predict Presidential Primary Elections

Web Science Project

Casey Flynn

Joy Reistad

Ross Moon

Contents

Abstract

There have been several instances of using data science in correlation with social media sites in order to infer outcomes of prominent social events (e.g. sporting events, stock prices, and even the overthrow of the Egyptian government (Widrich, 2012)). The use of Twitter data in correlation with data-science techniques has become common place in statistical modeling to predict the outcomes of the presidential primary elections. This topic, however is debated as to whether or not one can derive accurate results by merely mining uncensored micro blogs. The purpose of this project is to attempt to determine one way or another if the data contained in tweets can actually be used to predict the outcome of the presidential primary elections. By collecting over fifteen million tweets over the past several weeks this project demonstrates a strong correlation between Twitter data and the outcomes of the last two presidential primary elections.

Introduction

Elections years are a pivotal time in American politics as the outcome of the United States Presidential Election will undoubtedly shape foreign and domestic policy for the foreseeable future. Recently (Monday, Feb. 1 2016), the first primary election for the 2016 presidential candidates took place and we are seeing some of the narrowest margins for presidential primary candidates ever recorded ("Clinton ekes out win in Iowa against Sanders - POLITICO," n.d.).

During this event there was an unprecedented amount of user generated content created via social media. ("Twitter Moments: Sanders holds Clinton to virtual tie," n.d.). The information contained in the voter's tweets may very well have the potential to predict the outcome of the primary election.

If this theory holds true, we would like to analyze this data to create a model that will be able to accurately predict results of presidential primary elections before the final tabulation of votes. The consequences of being able to predict this outcome would serve as a model for predicting the United States Presidential election before polls close. The implications of this are astronomical. If we were able to inform otherwise abstaining voters that a party they directly oppose may win the election without their support, we could in fact shape the course of American history.

Relating to Web Science

Web Science is a broad field that represents the link between people and information through the World Wide Web and other online technologies. Discussion of web science topics has become common place in our everyday lives, and influential people including politicians agree that access to information is making the world a better place "the technological breakthrough of the World Wide Web has been enormously beneficial to society" – Mike Fitzpatrick, U.S. Representative, PA - R (Fitzpatrick, n.d.).

Methods Used to Gather and Visualize Data

Project Overview

The scope of the project was to create a process to accurately predict the presidential primary elections using data collected from social media. Several technologies were used to gather, process, and display the data collected from Twitter, many of which were discussed in CSCI 470 - Web Science. These topics include: Creating a persistent http connections; use of NO-SQL databases to house information; creating a web server to serve this information; and the use of web sockets to deliver information in real time to connected web clients. In addition to these techniques this project also employs the use of other web technologies not covered in class including sentiment-analysis, several JavaScript frameworks, source control versioning, and infrastructure provisioning.

To accomplish the project several tasks had to be completed. Firstly, to create a web application that scrapes social tweet data from Twitter relevant to the presidential candidates. Next, the tweet data must be deposited to a Mongo NO-SQL database. Then the data undergoes data analytics to predict the outcome of several Democratic and Republican presidential primary elections. Finally a web server must be constructed and serve the information in a meaningful way. The design of the project is visualized in the UML diagram seen in Figure 1.

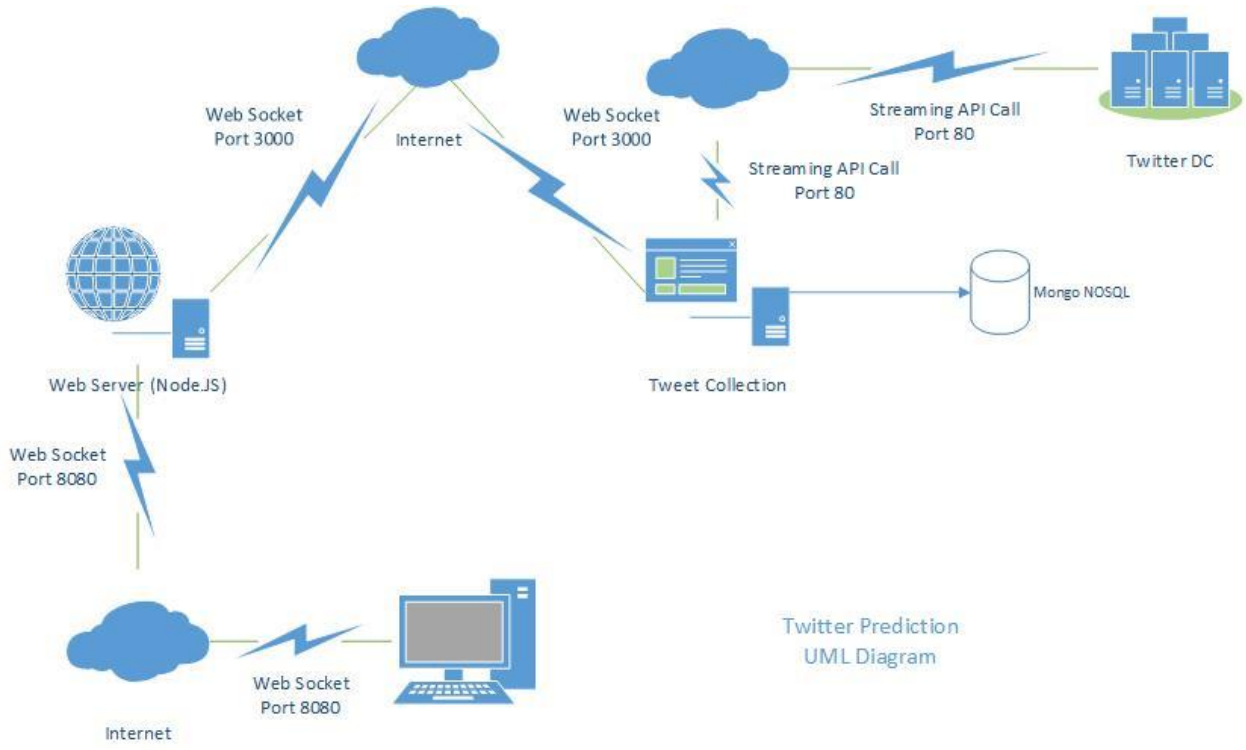


Figure 1: UML Diagram of Twitter Prediction Project

Data Gathering

In order to gather Twitter data, the project team leveraged the Twitter streaming API. The API offers low latency access to Twitter's global stream of tweet data without using a REST endpoint. To receive data from Twitter a persistent HTTP connection is established to the API endpoint. Whenever a tweet matching request criteria is made, specifically: a candidate's name was referenced in the text of the tweet, the endpoint serializes the tweet into a JSON object and sends it over the connection.

For the task of interfacing with the API endpoint an application was developed utilizing Node.js and publically available packages, "Twitter" and "mongoose", from the NPM repository. These packages are responsible for handling communication with Twitter and a Mongo database respectively. The rational for using Node.js was fairly straightforward; Node.js relies on an event loop that abstracts the need for a thread pool while handling requests. It also allows for asynchronous programming, while simultaneously offering a vast collection of user submitted libraries for preforming common tasks (including handling connectivity to Twitter, and Mongo).

The general program flow of our web application is as follows: The application first establishes a connection to Twitter. The application then utilizes the Node.js stream architecture; when data is received from the endpoint the application, it will asynchronously extract data from the incoming tweet (passed as a JSON object), serialize the data, and then attempt to store the information into the mongo NO-SQL database.

The application also hosts a web socket in order to provide the information to our web server which in turn may process the data and pass the information to connected web clients.

Volume of Data

Twitter is known for hosting an incredible volume of social data. There are "around 6,000 tweets are sent in an average second" ("Twitter Usage Statistics - Internet Live Stats," n.d.). This amounts to a around 518,400,000 tweets a day. The average size of a tweet is 2,500 Bytes (Valeski, 2011), which yields approximately 1.296 Terabytes/day of data. Millions of people (and their pets) are connecting around the world via Twitter.

While in service the web application collected data from April 19th to May 4th (Note: There was some downtime due to server issues). All data was inserted into a Mongo database, then subsequently offered as a stream to any connected client application in real time. During the aforementioned time span an impressive 7.019 GBs of tweets were collected in the Mongo database yielding a total of 17,287,694 tweets.

It should be noted that the Twitter API only allows a maximum collection of 1.0% of all tweets for free. This does not imply that our search criteria for the candidate names did not return all possible results; however, it cannot be proven that all possible matches to the search criteria were returned to the application. Meaning, that the collected twitter data is not a guaranteed one-to-one correlation the actual data. In order to receive all of the data from Twitter, a Twitter Fire Hose would be required. Unfortunately, Team Number Bee certainly doesn't have the resources to deal with that much data.

Web Server for Visualization

With mass amounts of data, it became essential to display information in a readable format. Proper visualization aids in the interpretation and readability of the data. To achieve this, four web pages were implemented. These page include: a real time tweet stream page, the preprocessed sentiment analysis for April 26th primaries, election prediction page for April 26th primaries, and real time sentiment analysis.

Node.js was used to implement the web server using the "Express" package offered on NPM. The Express package is a high performance robust tool for quickly creating single page web applications ("Express - Node.js web application framework," 2016). When a client connects to the web server, Express serves the client the web application in the form of a web page and associated JavaScript files. The other three web pages in the same matter, but the content differs for each page.

To access real time twitter data, the web server establishes a connection to the web socket offered by our tweet gathering application. It then constructs a web socket to communicate to connected clients. As a tweet is received, the text of the tweet is analyzed to determine which candidate names are associated with the tweet. Once the analysis is complete, it is offered to the client via the web socket.

Client Application for Visualization

The preprocessed sentiment analysis page for the April 26th primaries graphically shows the overall sentiment of tweets for each candidate during the week before this primary. This visualization is shown using the JavaScript InfoVis Toolkit, JIT. This is a tool used for creating interactive data visualizations for the web (Belmonte, 2013).

The election predication page is comparing the preprocessed data from the week before the April 26th primaries to the primary results. This page is a simply HTML which displays graphs that were generated from excel worksheets.

In order to render real time data on the client application React.js is used. React is a JavaScript library developed by Facebook that allows developers to create web components via a specialized templating engine. React relies on two in internal virtual DOMs to optimize performance, which allows efficient rendering of components. On load both virtual DOMs are initialized. If any changes are generated by React, then the virtual DOM React creates a delta of the virtual DOMS. Changes to the actual DOM are based on the delta of the virtual DOMS. ("React integration for ASP.NET MVC | ReactJS.NET," 2016).

Leveraging this technology, React components were created that will display tweets in real time, and display real-time statistics regarding the tweets with little to no noticeable latency on the client browser. The real time tweet stream helps visualize the volume of data the web server is collecting. It also graphically displays the tweets per second for each candidate. The real time sentiment analysis page analyzes the live feed of tweets with the Sentiment package. The data is categorized by candidate as a bar graph. Each bar is divided into subsections of positive sentiment, negative sentiment, and neutral sentiment.

Compiling the Application

In order to serve the client a working JavaScript package we must perform the following operations:

1. Use Browserify to prepare node packages to run in the client browser, leveraging jsx transform, es6 standards and babelify.
2. Compile react jsx components into a single JavaScript file.
3. Combine browserified packages, compiled jsx components and additional .js files into a single JavaScript file to be downloaded by the client.
4. Uglify and minify the JavaScript to obfuscate the code and compact it as much as possible for faster download.

Gulp is used to automate this work flow. Gulp is a task/build runner built to handle a development workflow ("gulp.js - the streaming build system," n.d.).

Analysis \ Findings

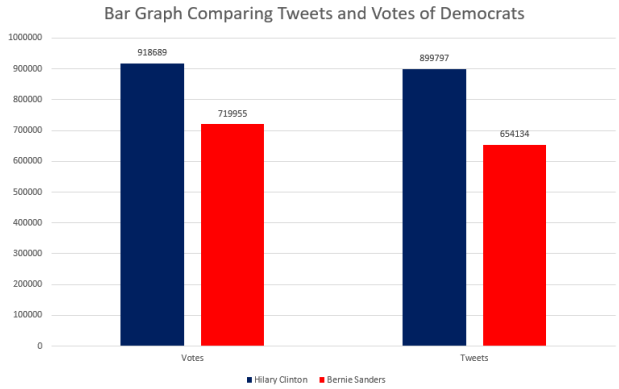
After gathering the tweets for the period we attempted to correlate the information. First actual vote counts for each of the five presidential candidate from Connecticut, Delaware, Maryland, Pennsylvania, Rhode Island were gathered. These counts were then combined into an overall count, and percentages were found for each of the four candidates.

These candidates were Donald Trump, Ted Cruz, Hilary Clinton, and Bernie Sanders. Due to an oversight during early stages of the project, twitter data was not collected for John Kasich. In hindsight this was unfortunate during the analysis stages of the project, because his actual votes had to be accounted for somehow. To overcome this oversight we evenly distributed Kasich's votes between Trump and Cruz based on the actual percentages of votes they received.

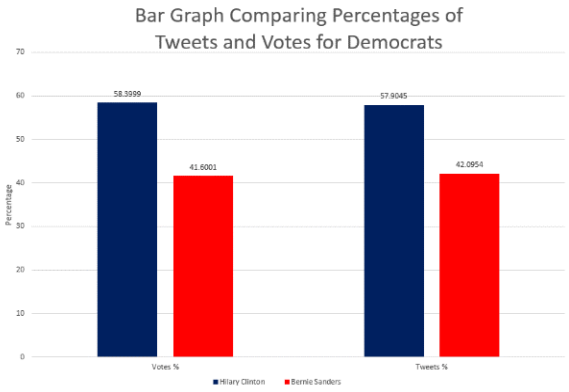
To analyze the sentiment of the data the project uses the package "Sentiment". The sentiment analysis algorithm used relied on the AFINN-111 word list. This is a list of 2477 English words each with a relative integer score ranging from negative five and positive five representing positive and negative words respectively. To determine the sentiment of a statement, we calculated both the absolute sentiment and relative sentiment score (absolute sentiment being the sum of all integer scores with 0 being used for words not contained in the AFINN word list, and a comparative score which is a the sum of the integer scores divided by the total count of words) (Nielsen, 2011). After completion of the analysis, it was found that the sentiment had little impact of the outcome of the election prediction. The volume of tweets had a higher correlation to election outcome.

Once this data was computed, we compared the vote counts and percentages for each parties' candidates to the tweet counts and percentages.

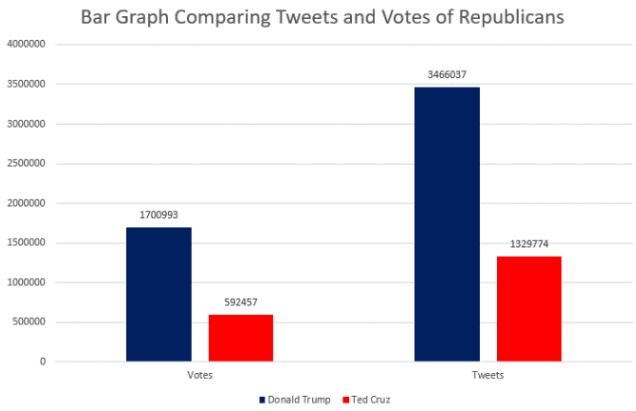
Findings



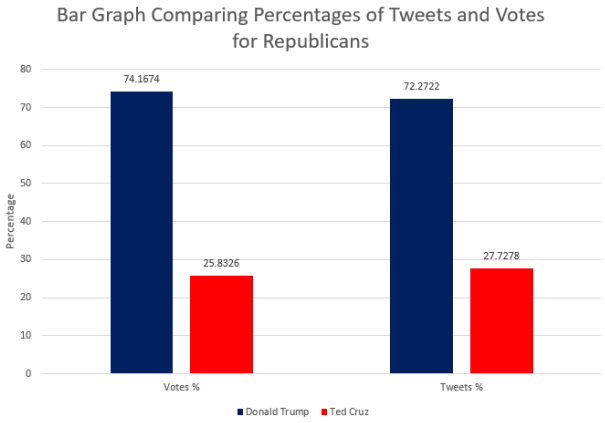
Graph 1. Democrat Comparison of Counts of Tweet vs Votes



Graph 2. Democrat Comparison of Percentage of Tweet vs Votes



Graph 3. Republican Comparison of Counts of Tweet vs Votes



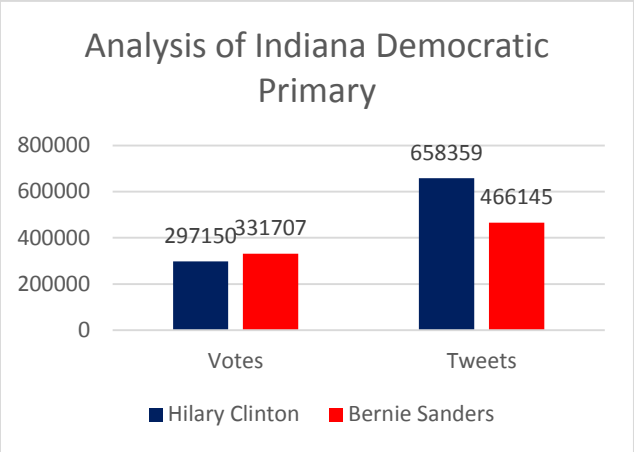
Graph 4. Republican Comparison of Percentage of Tweet vs Votes

Graphs 1 and 3 show the Democratic and Republican count comparison of tweets vs votes respectively. Graphs 2 and 4 show the Democratic and Republican percentage comparison of tweets vs votes respectively.

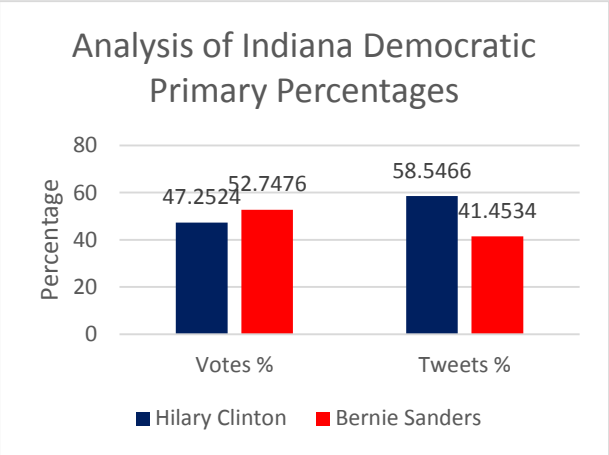
In Graph 2, the percentage of tweets for Clinton is 57.9%, while the percentage of votes for her is 58.4%. The percentage of tweets for Bernie is 42.1%, while the percentage of votes for him is 41.6%. In both cases, there is only an error of .5%.

In Graph 4, the percentage of tweets for Trump is 72.3%, while the percentage of votes for him is 74.2%. The percentage of tweets for Cruz is 27.7%, while the percentage of votes for him is 25.8%. In both cases, there is only an error of 1.9%. The vote percentages used in analysis were adjusted to account for Kasich votes.

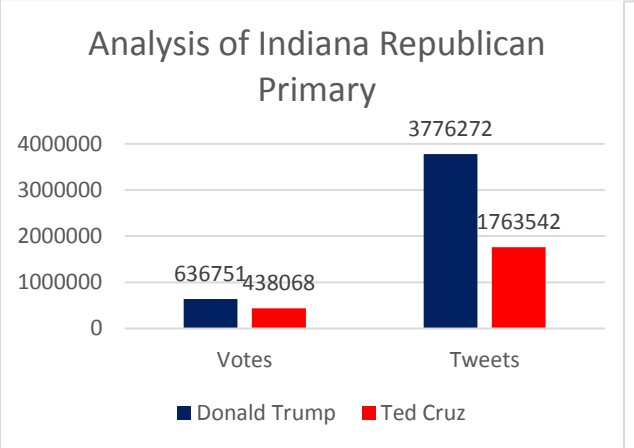
A second set of data was collected and analyzed for the Indiana primary on May 2. Due to lack of geographical data, the data could not be subsetted for just this area, and the results were not as accurately predicted.



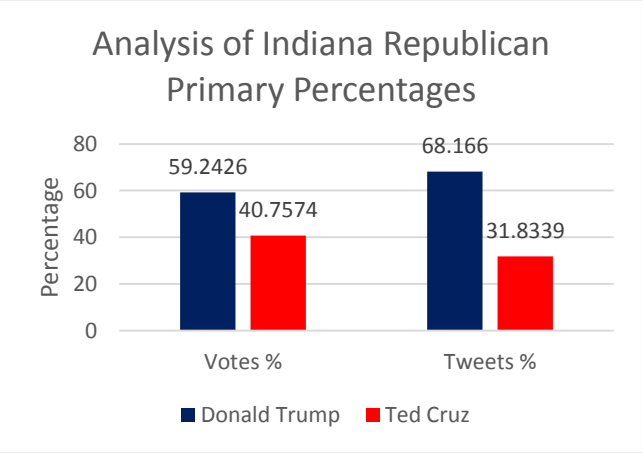
Graph 5. Democrat Comparison of Counts of Tweet vs Votes for Indiana



Graph 6. Democrat Comparison of Percentages of Tweet vs Votes for Indiana



Graph 7. Republican Comparison of Counts of Tweet vs Votes for Indiana



Graph 8. Republican Comparison of Percentages of Tweet vs Votes for Indiana

Graphs 5 and 7 show the Democratic and Republican count comparison of tweets vs votes respectively. Graphs 6 and 8 show the Democratic and Republican percentage comparison of tweets vs votes respectively.

In Graph 6, the percentage of tweets for Clinton is 58.55%, while the percentage of votes for her is 47.25%. The percentage of tweets for Bernie is 41.45%, while the percentage of votes for him is 53.75%. In both cases, there is an error of 11.3%.

In Graph 8, the percentage of tweets for Trump is 68.17%, while the percentage of votes for him is 59.24%. The percentage of tweets for Cruz is 31.83%, while the percentage of votes for him is 40.76%.

In both cases, there is only an error of 8.93%. The vote percentages used in analysis were again adjusted to account for Kasich votes.

The team speculates that the lack of accuracy in this prediction compared to the previous week's election is caused by the small number of votes cast on this date. The twitter data harvested represents a sampling from the entire world, not just Indiana. The accuracy of our prediction should have improved if more states had voted on this day. An alternative would have been to use only localized tweets, unfortunately this data was not accurately represented in most data collected.

Future Project Tasks

Currently the application is only available on a single web server. Given the proper time and resources secondary web server could be created. After the secondary web server is available a load balancing system would be added to the infrastructure to distribute requests between the web servers. This would allow for more simultaneous clients to be connected and also provide high availability should one of the web servers fail.

In retrospect, instead of a single instance of Mongo DB, the project should have leveraged a Mongo cluster to distribute the data collected over several machines. After implementing the cluster, the database would have been sharded then collections for each candidate would have been created (ideally one collection per machine, further sharding could split the individual collections across more servers if further scaling is required). This would have allowed us to maintain full text indexes for each candidate to do more analytics in real time.

Currently when attempting to build a text index on the data in the database, it will put enough load on the server to cease the collection of additional tweets. In order to process the data collected, scripts were developed to stream the output of general queries (no text search used) to an application running in node.js. The application then inspected the text of the tweet, and ran sentiment analysis, then loaded the information into separate collections. These smaller collections were then analyzed to produce or results.

Security was not a large consideration in the design of this project, since the information we had was public. The only security implemented was access to both the web and application servers. Moving forward it would be prudent to ensure all connections made by applications to the application server undergo some sort of authentication or use rule based firewall filtering. The next phase of the project would be to ensure all http connections, and web sockets be migrated to https connections.

Conclusion

In conclusion, twitter is a great source of information. The data from twitter can be used for many things, including election predictions. When used for election prediction, the percentage of tweets per candidate is quite accurate for predicting the result over a large area. It is less accurate when predicting the outcome in only one state.

There are many great tools for developing web applications. Over the course of this project, we were exposed to only a handful development tools. This project showed us that it was only the tip of the iceberg for the possibilities of web science.

References

Belmonte, N. G. (2013). JavaScript InfoVis Toolkit. Retrieved from https://philogb.github.io/jit/

Clinton ekes out win in Iowa against Sanders - POLITICO. (n.d.). Retrieved from http://www.politico.com/story/2016/02/iowa-caucus-2016-donald-trump-bernie-sanders-218547

Express - Node.js web application framework. (2016). Retrieved from http://expressjs.com/

Fitzpatrick, M. (n.d.). World Wide Web Quotes - BrainyQuote. Retrieved from http://www.brainyquote.com/quotes/keywords/world_wide_web.html

gulp.js - the streaming build system. (n.d.). Retrieved from http://gulpjs.com/

Kim Sang, E. T., & Bos, J. (n.d.). Predicting the 2011 Dutch senate election results with Twitter.

Nielsen, F. (2011). AFINN. Retrieved from http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010

React integration for ASP.NET MVC | ReactJS.NET. (2016). Retrieved from http://reactjs.net/

Twitter Moments: Sanders holds Clinton to virtual tie. (n.d.). Retrieved from https://Twitter.com/i/moments/694282991137329152?lang=en

Twitter Usage Statistics - Internet Live Stats. (n.d.). Retrieved from http://www.internetlivestats.com/twitter-statistics/

Valeski, J. (2011, July 29). data collection Archives - Gnip Blog - Social Data and Data Science Blog. Retrieved from https://blog.gnip.com/tag/data-collection/

Widrich, L. (2012, January 11). 6 Incredible Examples Of How Twitter Predicts The Future - The Buffer Blog. Retrieved from https://blog.bufferapp.com/6-incredible-examples-of-how-twitter-predicts-the-future