Twitter vs Trump: A Sentimental Approach

Munaf Arshad Qazi

Disclaimer

I do not have any affiliation with any political party. This study was conducted purely for educational purposes.

1 Introduction

2016 has been an eventful year, especially considering the POTUS electoral race. The media has been presenting it's side of the story while we've been hearing numerous times that President Elect Donald Trump is unhappy with the way he has been treated by media. The biggest surprise to everyone came when despite all odds (according to the media), Donald Trump won the elections showing he actually had a lot of support throughout the US. Since winning, PEOTUS has made a number of controversial decisions which have been heavily criticized/appreciated so when I heard him going back on his word of not pursuing Hilary Clinton Email Charges on the 22nd of November 2016, I wanted to see how the world reacted.

2 Approach

Twitter has two major APIs. A Rest API and a Stream API. The rest API can be used effectively for mining out tweets and is easily integrable into R, however with one drawback. To collect data, twitter has a 15 requests per 15 minutes limit [1], which in no way can be used to accumulate a million records without having multiple nodes with different keys making requests simultaneously. Therefore I decided to start writing a NodeJS script which would connect to twitter's streaming API and help me collect data in real time. The responses needed to be unique as well because identical tweets would just be added noise in my analysis and take up unnecessary storage space.

Once data was downloaded, I decided to filter out the fields of importance from it using map reduce. A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce tasks. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks [2].

Map Reduce can be applied to a number of different design patterns. Be it Filtering or Summarizations, Joins or Data Organizations all can be made simple by using such patterns. Following the book Map Reduce Patterns by Donald Miner [3] I was able to write a map reduce filtering script in both Mongo and Hadoop using Pig! Although I ended up using only the Mongo one because of my data was in Mongo aswell, both scripts are usable for the task except the data will need to be imported to HDFS for Pig. I have provided both scripts in the TMSA>scripts folder. Once map reduce was complete, using mongo scripts the data was exported into a CSV and ready for R analysis.

Polarity analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level—whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral.[4] For any such sentiment analysis, text needs to be converted in to numbers which a machine can understand. Thus lower casing, removing stop words, tokenizing all are processes which need to be done then. For the stop words, because this was for twitter feeds, which contains jargon and slangs, I created a stop words corpus of common words from the English language, Snowball corpus, terrier corpus and the minimal corpus. One can now gauge the polarity of text based on positive/negative dictionaries and create relevant plots. I used

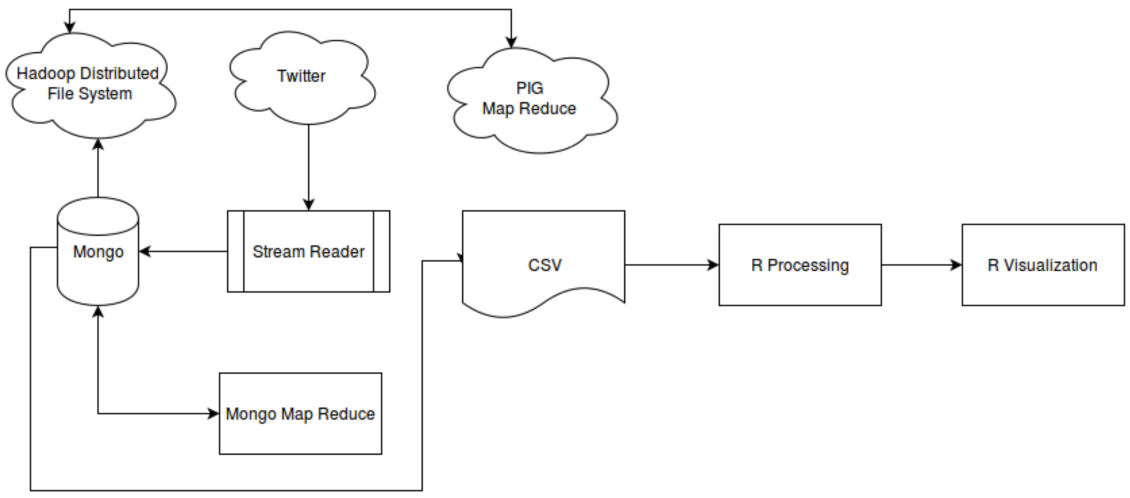


Figure 1: System Architecture diagram for Twitter Mining

a lazy learning **Bayesian classifier** to learn about sentiments from a learner list and then applied the classifier to the tweets and analyzed the results.

3 System Architecture

The stream reader connects to the twitter streaming API and reads tweets having 'trump' in their tweet_text. If so the tweet is downloaded and saved in a Mongo Collection. The reader has been configured to store tweets by their tweet_id so duplicate tweets aren't allowed in the Mongo Collection. The reader continues to read from the stream until 1 million tweets have been downloaded. This number can be re-set obviously.

Once this was achieved and data was collected, it needed to be prepared, the first part of which was filtering. For this project I decided to use Mongo Map Reduce to filter out my data easily because my main medium of storage was mongo again. Mongo offers flexibility in conversational data like twitter feeds and thus instead of porting data into HDFS for PIG, it could be directly manipulated in Mongo. However to avail cloud computing, the PIG implementation of Map Reduce for this filtering was scripted and tested as well and it is provided to see how performance varied. Once filtering was complete, data was imported into R for processing and visualization.

Figure 1 shows the System Architecture for the Miner. All remaining modules are explained in detail in the proceeding sections.

4 Data Architecture

The miner uses a NoSQL approach while storing data. This is because of the uncertainty of presence of some fields in the conversational dataset alongside the practicality of using NoSQL for a large amount of unstructured data. Because all these issues were addressed in Mongo and because of the flexibility and power mongo offers of manipulating documents and collections directly and not compromising on future extensibility, this miner was built incorporating MongoDB.

Data Set

The dataset contained more than **1.02 million tweets** collected over a 10hour period on the 22nd of November 2016 for the selection criteria 'trump'. In this huge set of data, I used tweet_id to maintain uniqueness and looked at tweet text for my sentiment analysis.
A sample response from twitter is shown in Figure 2 where the fields in the response are:

- _id - Mongo id of the tweet (same as tweet id for avoiding duplicates)
- created_at - date of posting

- id - id of the reviewer
- id_str - id of the reviewer in string
- text - text of the tweet
- source - link to the tweet
- in_reply - tweet object if in reply to
- truncated - whether truncated with other tweets
- in_reply_to_status_id - in reply to tweet id
- in_reply_to_status_id_str - in reply to tweet id in str
- in_reply_to_user_id - in reply to user id
- in_reply_to_user_id_str - in reply to user id in str
- in_reply_to_screen_name - in reply to user name
- geo - map location of the tweet
- coordinates - coordinates of the tweet
- place - user set location of tweet
- contributors - retweeted by whom
- is_quote_status - is a quote
- retweet_count - how many retweets
- favorite_count - how many favorites
- favorited - favorited by whom
- retweeted - retweeted by whom
- filter_level - request filters
- lang - tweet language
- timestamp_ms - tweet timestamp

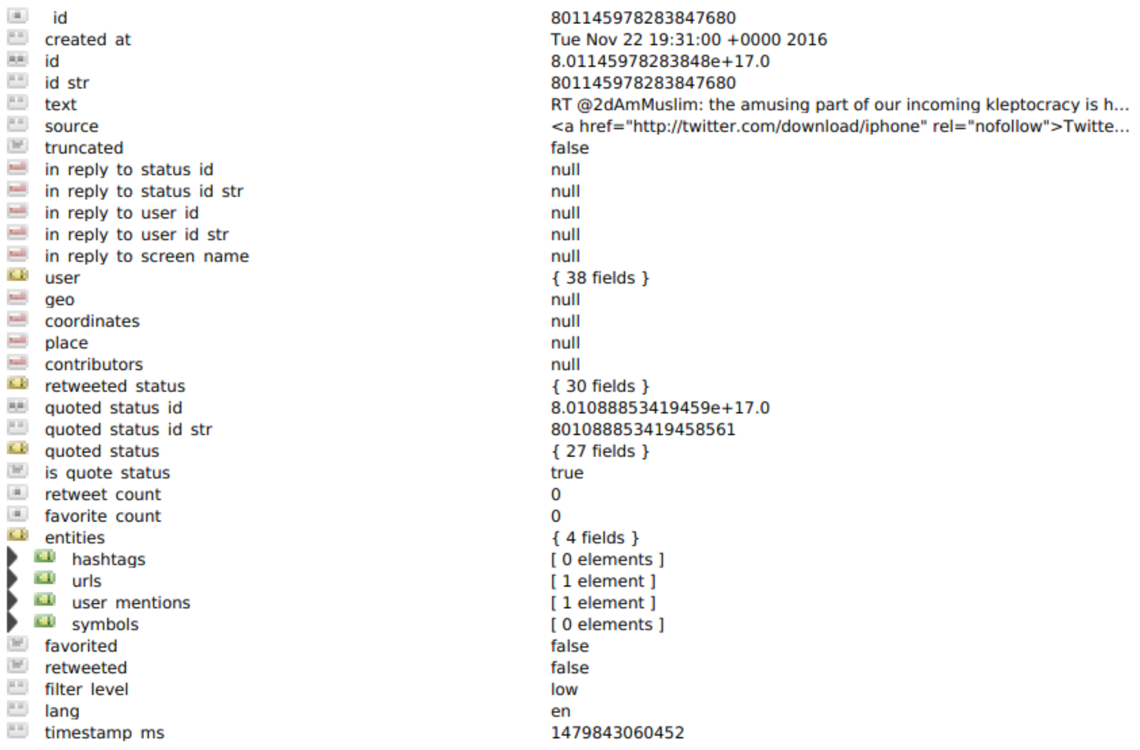
For polarity analysis, the program only requires the tweet id and tweet text, but considering the number of fields a response returns, an uncountable number of analysis can be thought of. To reiterate, because this is conversational data, it isn't necessary that all fields be present in every response. This is why using Mongo is pretty appropriate here.

5 Integration Architecture

The stream reader automatically connects to a running Mongo Server and creates the necessary databases and collections to store data into. Later MR can be directly applied to it, however if one decides to run PIG an extra step is required after creating a CSV and exporting to the HDFS. That includes adding zeros in place of nulls. Alternatives obviously exist such as using a mongo-hadoop connector[5]. Finally to connect the data in mongo to R, we export it out into a CSV so that a visual form of the prepared data is available external to the mongo server and the server doesn't need to be run every time processing in R is required. Again, this could be directly run inside R if an external copy wasn't required [6].

6 Design

Once data has been loaded into R, the data pre-processing and processing part can begin. A complete pre-processing and processing model is shown in Figure 3.



id	801145978283847680
created at	Tue Nov 22 19:31:00 +0000 2016
id	8.01145978283848e+17.0
id str	801145978283847680
text	RT @2dAmMuslim: the amusing part of our incoming kleptocracy is h...
source	<a href="http://twitter.com/download/iphone" rel="nofollow">Twitte...
truncated	false
in reply to status id	null
in reply to status id str	null
in reply to user id	null
in reply to user id str	null
in reply to screen name	null
user	{ 38 fields }
geo	null
coordinates	null
place	null
contributors	null
retweeted status	{ 30 fields }
quoted status id	8.01088853419459e+17.0
quoted status id str	801088853419458561
quoted status	{ 27 fields }
is quote status	true
retweet count	0
favorite count	0
entities	{ 4 fields }
hashtags	[0 elements]
urls	[1 element]
user mentions	[1 element]
symbols	[0 elements]
favorited	false
retweeted	false
filter level	low
lang	en
timestamp ms	1479843060452

Figure 2: Sample Response JSON by Twitter

Pre-Processing

Preprocessing is the first step for any Natural Language Processing System. All tweets must be uniformly formatted because the goal is for a computer system to analyze text. Because of a computer's inability to comprehend the significance of punctuation marks and capitalizations, symbols and digits and because of their low importance in opinionated texts, we remove them. Tokenizing ensures all sentences are processed as one row of a matrix with each word an element.

1. Remove 'retweeted via' and symbols
   The first step is removing all symbols in the tweet along with the RT|via tag for retweets. Note here, that a retweeted tweet and the original tweet have the same twitter_id, thus only one copy is present in the collection (no duplicates).

2. Remove all punctuation marks, digits and HTTP links

3. Remove tabs and extra spaces

4. Convert to lowercase and remove all nulls

5. Remove all stop words
   For this miner, I created a stop words corpus manually from the snowballC corpus, terrier corpus, minimal corpus and the standard English language corpus, as well as adding common political terms in Donald's speeches which would be of little polarity value such as 'China','Mexico','Muslims','Ivanka' etc.

6. Convert into a matrix
   This is an important step because R will later evaluate a Document Term Matrix from this matrix.

Processing

This part is where a magic happens. A Naive Bayesian Classifier lazy learns about subjectivity in words, and then applies the learning to the processed tweets. It then outputs the result into a CSV.

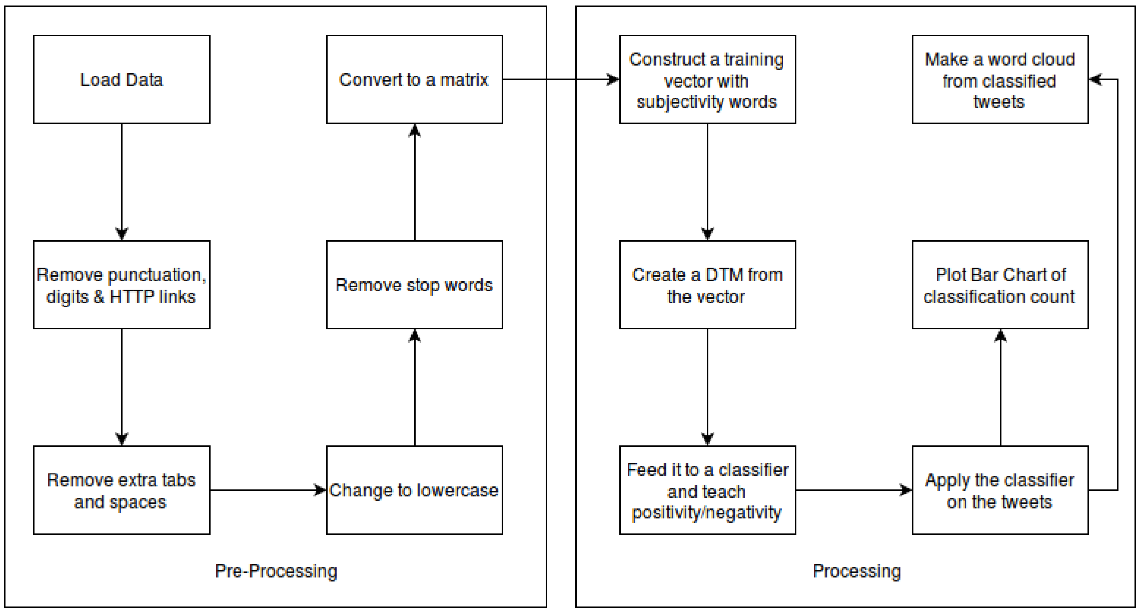

Load Data
Convert to a matrix
Remove punctuation, digits & HTTP links
Remove stop words
Remove extra tabs and spaces
Change to lowercase
Pre-Processing
Construct a training vector with subjectivity words
Make a word cloud from classified tweets
Create a DTM from the vector
Plot Bar Chart of classification count
Feed it to a classifier and teach positivity/negativity
Apply the classifier on the tweets
Processing

Figure 3: Complete R Processing Model

1. The subjectivity words are from a subjectivity list joined together to make a training column. This list contains common English terms used for deciding the positivity or negativity or texts.

2. Document Term Matrix
   A DTM is created from the vector source of the training column. A document-term matrix or term-document matrix is a mathematical matrix that describes the frequency of terms that occur in a collection of documents. In a document-term matrix, rows correspond to documents in the collection and columns correspond to terms.

3. Learning
   The classifier learns about words and values from subjectivity.csv. This is a naive bayes classifier that learns from the original training corpus how to classify the three categories found in the corpus: Positive, Negative, and Neutral. Naive Bayes was used because such classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. Because this is a maximum likelihood problem, a naive bayes solution is one of the best candidates out there.

4. Classification
   Using what was learnt from the subjectivity training set, one by one, all tweets are analyzed for polarity and a value judgment is made upon the text of the tweets.

5. Results
   They are saved in a data frame along with category and polarity.

7 Validation

Once the polarity results have been generated, the tweets and polarity data frames can be manually inspected to validate the results. My inspection showed me results such as the ones shown below:

Negative

```
* Trump deserves credit for breathing and expelling words.
* RT @Cernovich:  I said on @mitchellvii's show before the election
```

Polarity	Number of Tweets
Positive	211569
Negative	575712
Neutral	241476
Total	1028757

Table 1: Final Polarity Results

```
that Trump wouldn't prosecute Hilary.
```

Positive

```
* RT @Nigel_Farage:  '@realDonaldTrump's popularity soaring in the
US. Excellent talk to the nation yesterday.
* RT @PrisonPlanet:  Trump to accomplish more for low income
Americans on day one than Obama did in 8 years.
```

Neutral

```
* RT @nytimes:  Donald Trump has finished meeting with Times
reporters and editors.  Here's a look at what he said.
* RT @educationweek:  K-12 Under Trump:  Who wins and who loses?
```

These are some good examples of the ones classified by the miner. Please note that to maintain decorum, I shared only decent negative and positive tweets above.

8 Results

The total number of tweets analyzed with polarity is shown in the Table 1. I'll just mention the results generated in this section, and analyze them in depth in the analysis section.

Plots The tweets would be more comfortably visualized in a bar chart. The final bar chart plot obtained is presented in the Figure 4. From a bird's eye view, according to twitter, the evening of 22nd November wasn't particularly a high point in Trump's political career.

Word Cloud A word cloud was generated for all tweets identified with their polarity. This can help understand how people are feeling about the different things Donald Trump has been talking about. The word cloud is shown in Figure 5.

9 Analysis

To analyze the results, lets look at each item individually.

Table It is interesting to see that within a 10 hour span, an individual can generate so much traffic on social media, and can stir up emotions of the world with a few statements. With his decisions about the housing secretary [7], not pursuing the Clinton email case [7] and meeting with Top TV anchors[8] he was able to generate more than 570k reactions against him. It must be noted that he did have around 21% supporters backing him up as well which amounts to another large figure of 211k.

Plots The bar chart in Figure 4 is a good depiction of the table in Table 1. It can be easily seen that on twitter, haters out number Trump's supporters by twice the figure. Because Donald Trump went to win the US 2016 Presidential Elections, these two facts can be used to infer where the twitter crowd belongs to geographically.

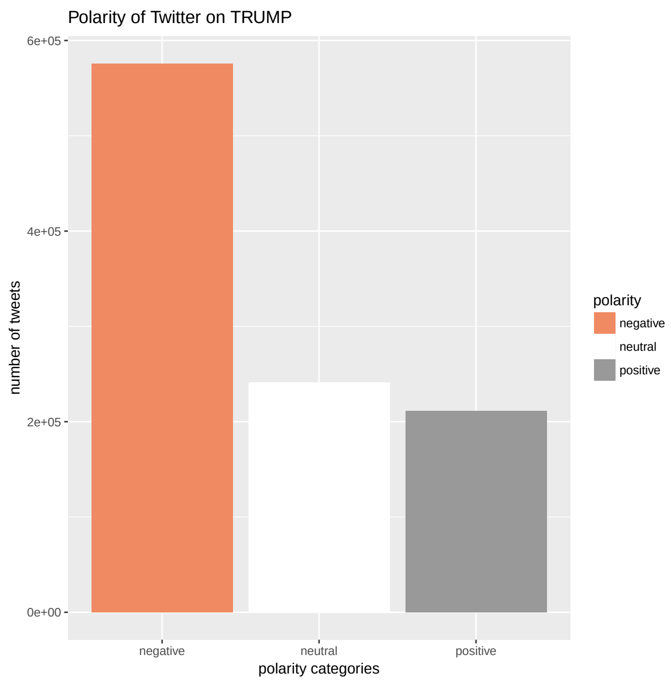


Figure 4: Bar Graph of Polarity

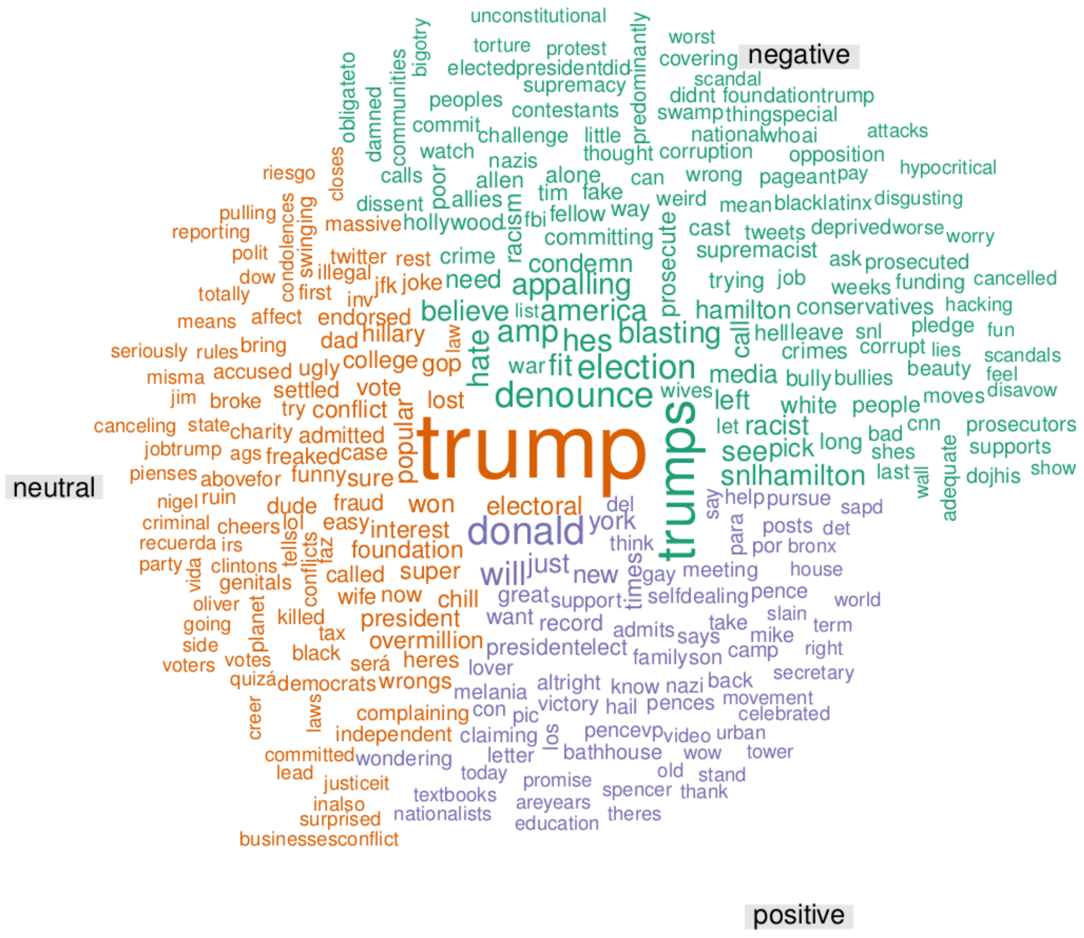


Figure 5: Wordcloud of terms about Trump along with Polarity

One of my analysis is that because twitter has been despising the PEOTUS clearly, it is used by people much more of a democratic alignment. Transitively, a majority of these users belong to the coasts rather than central America because that is where the democratic vote bank was these elections. On the other hand a majority of voters for Trump came from center. This hypothesis can only be verified if we analyze twitter usage patterns in the United States.

I decided to construct my own map from the tweets in my dataset but I soon realized that the conversational data that I was recording using the streaming API, didn't record locations for every tweet. It depended on the privacy settings set by the user. Therefore I looked at studies on American twitter usage. Consider the Map in Figure 6 taken from a study by Smithsonian.com [9]. Next look at the map, in Figure 7 by thoughtfulreading.com [10] which shows how the election results varied by county.

This is perhaps one of the most fascinating revelations of this study. It becomes vivid that the twitter users in the US directly correlate with the democratic vote bank. Making it ascertain that any Twitter based study on the PEOTUS will definitely be biased against him. Even more interestingly, it is possible that the democratic campaign was more geared around social media particularly twitter compared to that of the republicans. If so, Hilary definitely ran a successful campaign on twitter winning almost everywhere she had access through it. The power of social media never ceases to amaze.

Word Cloud Observing the word cloud, we can see a lot of common English terms have been removed. Trump was the most widely used term throughout (which makes sense because this was the search criteria on the streaming API as well). From the other words in their we can draw a number of other inferences, the highlighted words are from the word cloud.

A lot of negative tweets talked about Trump *denouncing* his promise about Clinton email charges. There were tweets referring Trump as a white *supremacist, racist, scandalous* and a *bully.* People *condemned* the elections and called it *torture.*

While Trump supporters were positive about his *secretary* choices, about his *education* reforms. They were happy about the *victory* and referred to him as a *lover* and *just* person bringing a new *movement.*

Tweets which were found neutral usually mentioned how trump is *broke*, how the whole scenario is *funny*, some passed on *condolences* while others accused *voters* for the outcomes. It is interesting to see that neutral tweets also talked about *Hillary* Clinton and the *Democratic* Party.

There are a number of other ways this cloud can be interpreted. For e.g it can be used to understand how a specific group of voters feel about the President Elect and can be actively used to target those areas and boost his approval ratings. It can also be used by the political parties to spread fear amongst the masses about topics people already are concerned about.

10 Operating Instructions

A complete set of step by step operating instructions have been provided in the project directory inside Instructions.txt.

11 Performance

Considering the large amount of data to download and future processing to entail, it was absolutely a must that no duplicates be recorded. Along with that the pre-processing and data preparation needed to be done effectively as well so as to minimize the time wasted in these processes. This is because teaching a classifier and applying classification is a time consuming process. To support this task's efficiency as many stop words as possible were to be removed from the tweets alongside all extra unmeaningful characters. This is why a special stop word corpus was designed using 4 different corpus'.

12 Conclusion

Twitter is the gold mine when it comes to looking for opinions of people on any topic because of its popularity and customer base. On average Twitter generates 500 million tweets in a single day!

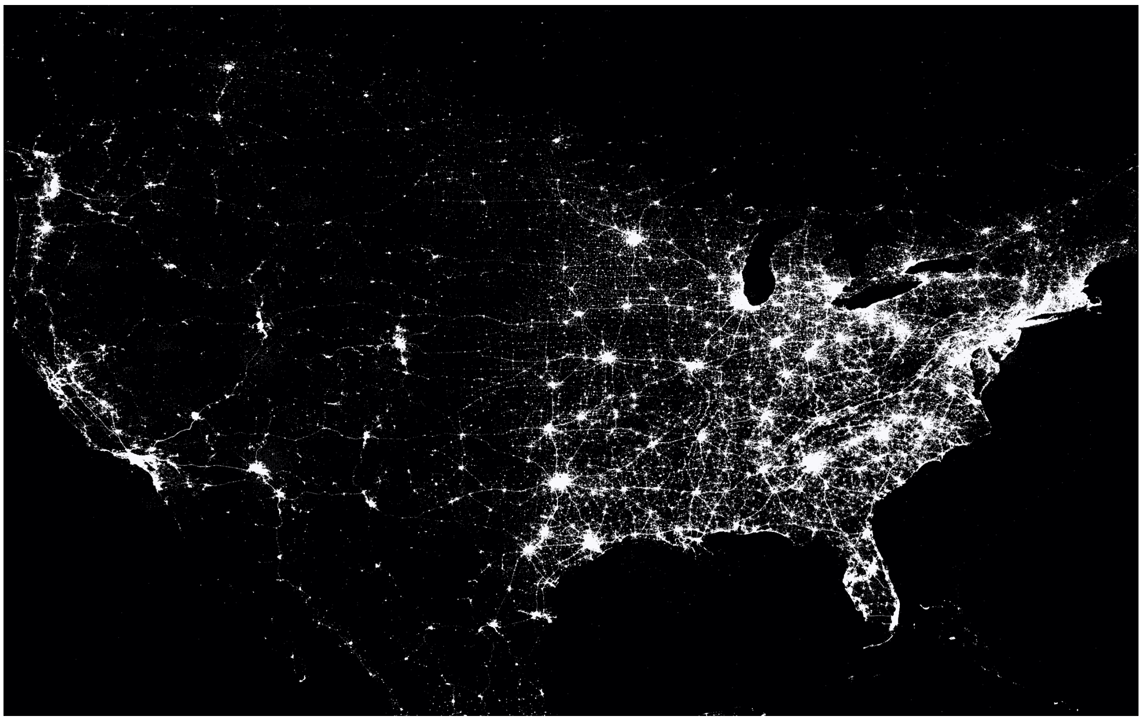


Figure 6: Map of Twitter Users in America

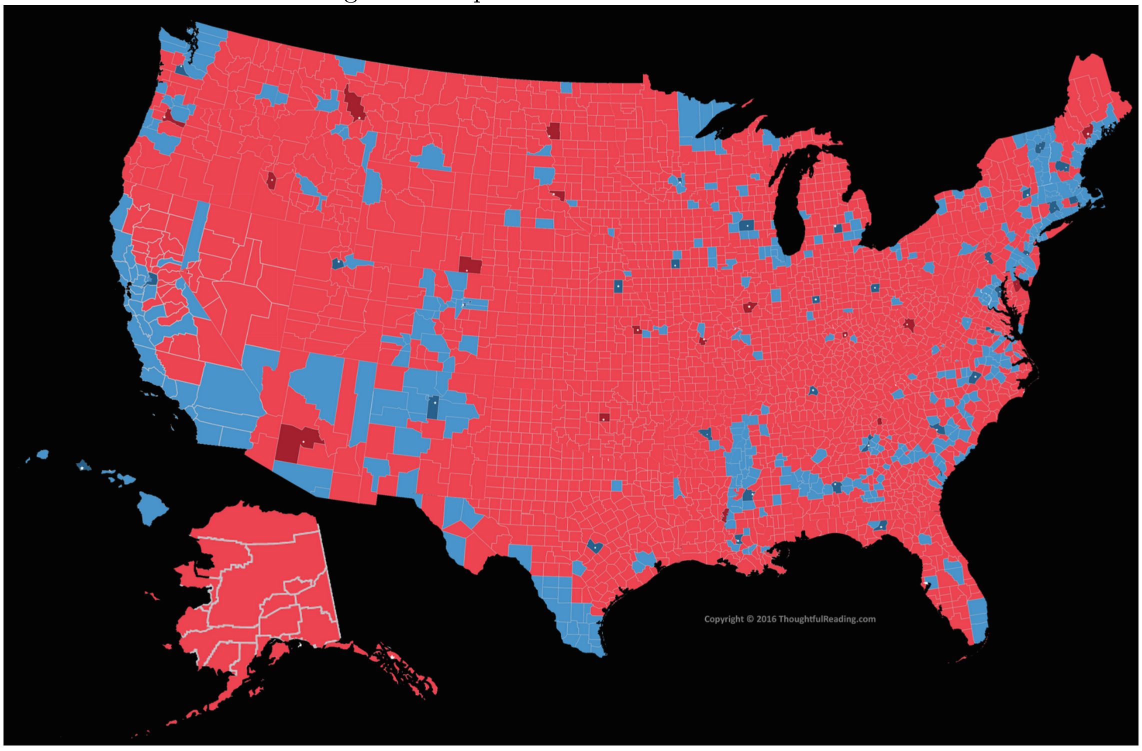


Figure 7: US 2016 Election Results by County

[11] Using it I was able to extract what people think about the new President Elect on his planned reforms and policies. I analyzed 1 million tweets relating to Donald Trump.

Results showed a huge amount of negativity for him, but considering he won the presidential race, I went on to look at the map of Twitter users in America. It became obvious from there that twitter is being used to voice opinions by a specific group of people which don't make up a big part of his vote bank. His supporters, mainly in the inner regions away from both the coasts of America, aren't very active on the social medium.

Next, it was helpful to look at the word cloud generated using the tweets related to the PEOTUS to better understand the people's perspective. The negativity usually was around his pre-election day claims and statements, while his supporters lauded him for the movement he planning to bring. The neutral tweets included some about about the democratic party. All in all, this was a fun and fruitful exercise to learn about the political landscape of twitter users in America.

I would like to acknowledge the efforts of Dr. Raman Kannan (NYU) for his constant motivation, as well as Twitter for opening their hearts as well as their sockets for an unlimited number of incoming requests through the streaming API. Finally, IBM for providing a platform which kept my interest in Big Data and Cloud Computing invigorated and exposing me to state of the art technologies in use these days.

References

[1] Twitter API Rate Limits. *https://dev.twitter.com/rest/public/rate-limiting*

[2] Hadoop. Map Reduce Tutorial. *https://hadoop.apache.org/docs/r1.2.1/mapred_ tutorial.html.*

[3] Miner. Donald. *Map Reduce Patterns http://barbie.uta.edu/ jli/Resources/MapReduce&Hadoop/MapReduce%20D*

[4] Sentiment Analysis. Wikipedia. *https://en.wikipedia.org/wiki/Sentiment_ analysis.*

[5] Mongo-Hadoop. *https://github.com/mongodb/mongo-hadoop.*

[6] Getting Started with MongoDB in R. *https://www.r-bloggers.com/getting-started-with-mongodb-in-r/.*

[7] New York Times. 22 Nov 2016. http://www.nytimes.com/2016/11/22/us/politics/donald-trump-transition.html?_

[8] Democracy Now. 22 Nov 2016. https://www.democracynow.org/2016/11/22/headlines.

[9] New York Times. 22 Nov 2016. http://www.smithsonianmag.com/science-nature/the-world-according-to-twitter-i

[10] US 2016 Election Map by county. http://thoughtfulreading.com/politics/live-map-united-states-2016-presidential- Thoughtful Reading.

[11] Twitter Usage Statistics. http://www.internetlivestats.com/twitter-statistics/. Internet Live Stats.