Mining Trends for 2016 Election from Twitter Data

Sili Hui Ayush Jain Anirudh Jayakumar

Department of Computer Science
University of Illinois at Urbana-Champaign

ABSTRACT

2016 is the year of United States presidential election. As of now, presidential candidates have started countless campaigns and rallies, in the meantime discussions of such political issues also took place over the Internet space, such as Facebook, Reddit, and Twitter. To get in depth knowledge of what most people think of each candidate, it is hard to get plausible data from the former due to high selection bias. The latter, Internet space, on the other hand, provides both trust-worthy data source and possibility of mining opinion. In this project, we take advantage of such idea and present trend analysis of presidential election based on Twitter data by utilizing a specialized model called VADER [8].

Keywords

text mining, Spark, sentiment analysis

1. INTRODUCTION

One of the hottest topics now in United States is the 2016 presidential election. Countless discussions are going on every platform, making it almost impossible to digest all information. Especially, for those who don't follow the process closely, a natural question to ask regarding this topic is that *how many people are supporting each candidate and what their poll look like from time to time.*

To answer this question, there are two major source of information, namely media source and poll from website. Major medias are polling opinion from the public once a while and report their observation by surveying selected audience. Such method to demonstrate the support of each candidate has been used for decades, yet the it is both time-consuming and inaccurate due to its nature. Having people filling out survey is itself a time-demanding job. By the time the poll is aggregated, the real statistics may have already shifted to a different state. Another nature of this method is it is prone to selection bias. Polling from selected people may not represent the overall picture of the real statistics. The other method, online polling, such as [4], suffers from the same drawbacks. In addition to these two drawbacks, such sources of information is often times not well-organized. They either do not show each candidate's individual statistic or visualize the trend is a not-user-friendly manner. For example, if I want to know Donald Trump support rate and trend in Feb, it would require some effort for me to figure out how he was doing exactly. Therefore, these sources of information is not optimal for our problem.

According to a study in [7], there are approximately 56.9 million active users on Twitter in 2016, which is approximately 18% of the whole population of United States. Many of them express their opinion towards the presidential election on Twitter every single day. However, there is little-to-no existing solution built on twitter data. With that, we believe that if we can utilize Twitter data, we can get a novel system and have a solid understanding of the election process.

Thus, the main goal of our project is defined as following. We aim to serve as a tool for those people who don't follow the 2016 election closely. We will take advantage of Twitter data and perform trend analysis about 2016 presidential election. We will be analyzing what Twitter users think of each party and each candidate over time. If geo-location is given, we will also be analyzing the distribution of opinion for different regions and states. Potentially, we can also predict the future trends based on current data. In addition to the batch process we build from collected data, we also want to support streaming functionalities. In the end, we hope users to our system can have a better knowledge of the 2016 election as a whole and its trends, both historically and in real-time.

With this aim in mind, the question comes to how to make it happen. As we build our system, there are couple challenges we need to address.

Challenges

1. Users are expressing their opinion on everything on Twitter:

They could be expressing towards sports, flight delay, new deals and etc. Among all their tweets, only small portion of them are related to the topic we care, the 2016 election. The majority of other tweets does not help to the analysis.

2. Internet language is not a formal language:

Twitter systems does not check spell and grammar error whenever a users tweets. Thus, a decent amount of the data on Twitter actually cannot be interpreted exactly by many natural language processing toolkit.

3. Volume of the data:

As is mentioned earlier, Twitter has a large amount of users and produces a large volume of data each day. Often times, such volume of data cannot be stored in a single machine. Thus, if we need to implement a module that leverage Twitter data we need to make such module scalable.

4. Sentiment Analysis on Tweets:

To analyze the breakdown of people's opinion towards each presidential candidate, we need to do sentiment analysis on each tweet, which is a hard problem itself in natural language processing. The complexity of the problem gets

even worse when it involves analyzing sarcastic tweet.

Due to these challenges (especially 2 and 4), our goal is naturally hard to achieve without modifications. Thus, to get a decent approximation of our goal, we made the following assumptions:

1. most of the tweets that contains informal language does not affect the big picture overall

2. sarcastic tweets are merely small fraction of overall tweets

After weeks of discussions, we have decided adopting many open sourced projects into our project. Specifically, we use *Spark-streaming* [2] and *NLTK* [5] in our project.

In the following sections, we will dive into details of our project. We will first describe collected dataset and how do we collect data in Section 2. Section 3 will cover our architecture design and implementation details. We will talk about our observations and results in Section 4. In section 5, we will discuss what we have learned and future work.

2. DATASET

We will first talk about how to collect data, and then talk about the data we collected.

2.1 Data Collection

The task of collecting Twitter data itself is not as hard as it seems. The flow of crawling process can be found in Fig.1.

To get Twitter data, we need to first register a Twitter App. Then, we use the app credential to collect data from the streaming API. The streaming API is the public api exposed by Twitter that allows developer to collect certain percept of tweets that flow through their system in real time manner. This api itself is exposed as REST API and all communications are done in the format of HTTP request. After sending the request object to API, we get a series of JSON objects back.

Fig.2 shows an example of the JSON object we get back from the API. It contains all information of a tweet.

2.2 Collected Data

Due to the volume of data itself and other constraints, we didn't have large enough corpus collected at the time being. Thus, in this project, we will mainly use the Twitter data collected by Professor Chengxiang Zhai. Professor Zhai has collected Twitter data from 2015 to now, and we get his permission to get a copy of desired data. For the specific topic of our project, we will be utilizing Tweets data from time beings between Feb 2016 and Mar 2016, the period when Congressional and State Primary election took place.

As we have mentioned in previous section, not all tweets are related to the election topic. To get those related tweets, we applied a layer of filtering on top of those twitter data. That is, we assume tweets that are related to the topic of election are those satisfy the following properties:

- contain related hashtags in the tweets
- tweeted from the candidate twitter account
- contain name of the candidate
- contain tags that are related to political issues

After such layer of filtering, we get a much smaller size of data. All the following analysis is based on this smaller dataset.

Fig.3 shows the list of keywords we used to filter desired tweets. One thing to note is that we also include hot political topics during the process of filtering.

In summary, we will using Twitter data of total size 275GB collected in during Feburary, March, April of 2016. After filtering, only 5 GB of data are related to the election topic.

Figure 1: Flow of Twitter data crawl

Figure 2: Sample Tweet Object

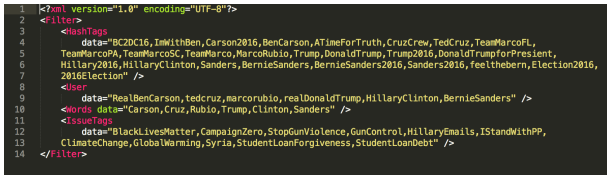

Figure 3: Keywords used for filtering

3. ARCHITECTURE AND IMPLEMENTATION

We will cover our architecture and implementation details in the following.

3.1 Big Picture

One challenge that arise during the design phrase, as we mentioned in previous section, is the volume of data we will be handling. Specifically, there are 275 GB of tweets in total, and we need to filter out undesired data from useful data. The complexity of the data can get even worse if it is implemented as real-time streaming.

To handle the complexity of such data, we need our design to have the capacity to store and process such large volume of data while keep the overall design as simple at the same time.

Fig.4 shows the pipeline flow of our data. We support two types of data sources, the source from the real-time streaming API by Twitter and the collected data. After data is fired from the source, it will first go through filtering stage to filter out none-relevant tweets. The idea of this stage is described in Section 3.

After the filtering stage, we pipeline the data through the text analysis stage, the stage where we append indicators (usually tags) to the data that tells the sentiment of this tweet. The details of this stage will be discussed in the following section.

After the text analysis stage, the tagged tweets will then be pushed into destined storage for aggregations purpose. One thing that needs mention is that Aggregation can be done either in disk or in pipeline, where we strip away non-necessary content and only keep the aggregated results in memory that can be later used as output or presentation.

The final stage is the presentation. The aggregated Due to our lack of expertise on GUI, we implement the presentation in a batch format. Namely, we will generate plot and trends as images and store them in local disk. For the collected files, we generate plots corresponding to time-line. For the real-time tweets, we generate plots by fixed interval.

3.2 Sentiment Analysis

Sentiment analysis, as we mentioned previously in challenges, is generally a hard problem. In lecture, we learned mainly two methods for such task: the ordinary logistic regression and latent aspect rating analysis [9]. For this task, we decided to use neither methods for two reasons. Firstly, many existing approaches are evaluated on datasets that hardly share anything with our data, thus there is no strong guarantee on accuracy. In other word, a more social media-focused approach will yield better accuracy given the diversity of Twitter data. The other reason for us not choosing these two methods is the over-complicated feature extraction work on tweets. By all means, every single tweet has only a limited amount of characters with small amount of contextual information. Not only we can only propose limited features, but those classical classifiers may be extremely sensitive to the features we feed in. Thus, we will need careful data engineering and feature extraction work to make sure we are using reasonable features. On top of these constraints, we also could not find a open-sourced implementation of latent aspect rating analysis.

After additional studies, we decided to use the VADER model [8] that is built specifically for social media text. VADER stands for Valence Aware Dictionary and sEntiment Reasoner, and it is a rule-based sentiment analysis tool that came with NLTK package.

The output of VADER model is a series of probability of whether the input text is positive, negative, or neutral. Fig.5 shows examples of what the output looks like from VADER.

3.3 Implementation

Bearing all those concerns in mind, we made the following design choice.

We decided to use **Python** as the main language that handles everything from data crawl to the implementation of all stages for 2 reasons. Firstly, its nature of easy-to-use-ness and compilation in runtime dramatically reduce the development cycle for this project. We could use Java instead, but it would require way more code to achieve the same performance. Secondly, many existing packages have Python support. It will achieve better performance if we use those packages.

For the pipeline framework, we decided to use the **Spark Streaming[2]** for the following reasons. Firstly, we want to learn a new scalable computation framework. Hadoop[1] also works, but it is mainly designed for batch processing and does not have good support for real-time processing. The design of Spark, on the other hand, is heavily emphasized on in memory process and streaming. More specifically, it has official support of Spark Streaming which performs just like Storm. Another reason we choose Spark is that Spark has perfect Python support as well, which fits perfectly to our needs. Storm[3] is another replacement for our purpose, but it does not have good python support so we did not use it.

For the sentiment analysis stage, we decided to use NLTK[5] package that embeds VADER model we discussed earlier. We will feed each tweet that piped through this stage into VADER model, and attach tags generated from VADER to the tweet indicating the sentiment of this tweet.

The final presentation will be plotted using Plotly[6], a online tool that has nice python support for plotting.

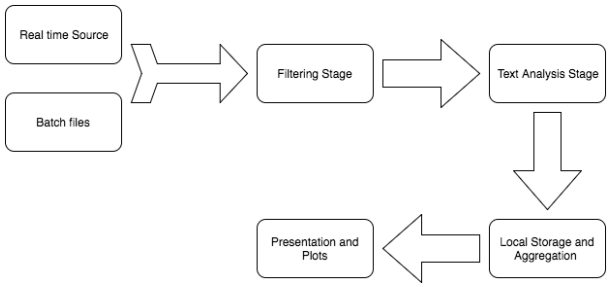

Figure 4: Data pipeline

```
The book was kind of good.
compound: 0.3832, neg: 0.0, neu: 0.657, pos: 0.343,
The plot was good, but the characters are uncompelling and the dialog is not great.
compound: -0.7042, neg: 0.327, neu: 0.579, pos: 0.094,
```

Figure 5: VADER Example Output

4. OBSERVATIONS AND TRENDS RESULTS

In this section we look at some of the trends we observed from analyzing the twitter data. We specifically look at popularity of candidates and also the sentiment the public has towards them. Then we look at public sentiments in specific states (New York and Texas) and see how these sentiments translates into votes in the presidential primaries.

4.1 Popularity of Candidates

Fig. 6 shows the popularity of presidential candidates over the course of 6 weeks from mid February to mid April. As expected, Donald Trump is the most popular candidate in this election. He has run a very controversial campaign and has attracted huge attention from the media and general public. It is important to also understand that popularity doesn't always mean positive sentiment. The other republican candidate Ted Cruz is second in the popularity charts. This is a bit of a surprise since his campaign has not been

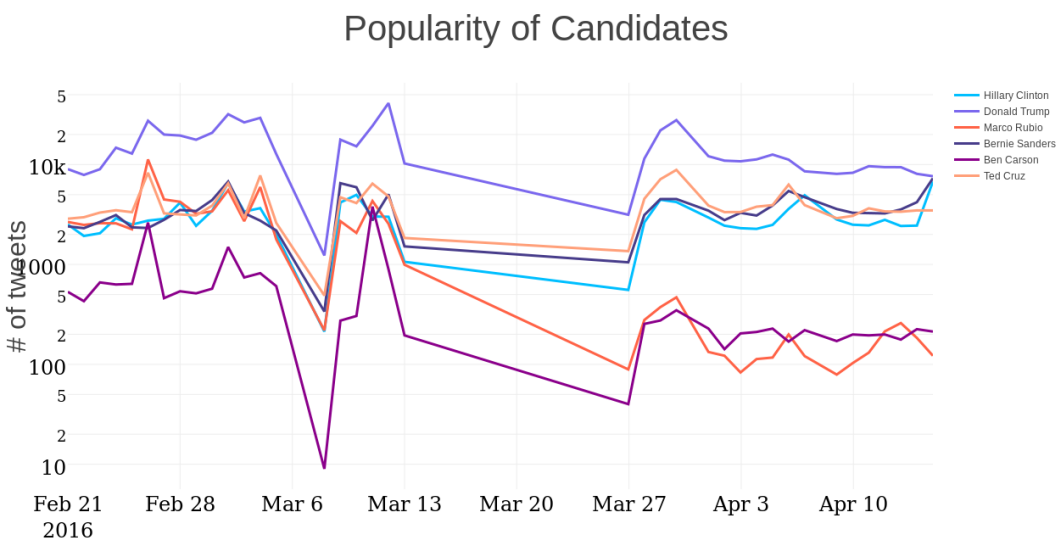

Figure 6: Popularity of Candidates

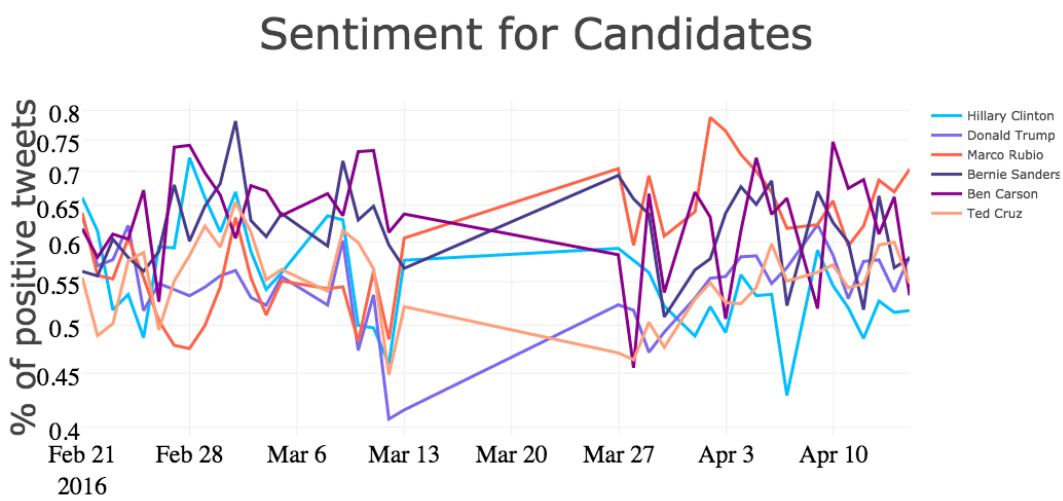

Figure 7: Public Sentiment towards Candidates

very controversial. From the democratic party, both Clinton and Sanders have similar popularity with Bernie Sanders having a slightly higher popularity. This could be a result of Bernie's huge popularity among the young voters.

4.2 Sentiment over Time

Fig.7 shows the sentiment of public towards the presidential candidates. The republican party candidates Donald Trump and Ted Cruz have lesser number of positive tweet percentage even though they have higher popularity. Between them, Mr Trump has lower percentage of positive tweets. This is not surprising due to the ultra-aggressive style of campaigning Mr Trump's team have been doing. In the democratic side, Bernie Sanders have more percentage of positive tweets than Hilary Clinton. Bernie has run a more issue based campaign with a lot of support from young voters. This could be one of the reasons for more positive opinion about Bernie Sanders' campaign over Hilary Clinton's. One of the interesting aspects to see is if these positive sentiments will result in votes and victories in the primary elections.

4.3 Location based Sentiments

Only a minor percentage of the tweets had location information. Therefore the following results may not be very accurate due to the smaller sample size. In our analysis we look at the sentiment in two states – New York and Texas. We first compare the sentiment towards republican candidates in New York and Texas. Fig.8 and Fig.9 shows the public sentiment towards Donald Trump and Ted Cruz in New York and Texas respectively. In New York, Ted Cruz has a small but consistent lead over Donald Trump but there is more fluctuation in Texas but Mr Cruz still has slightly better sentiment rating. Now, let's move to the democratic

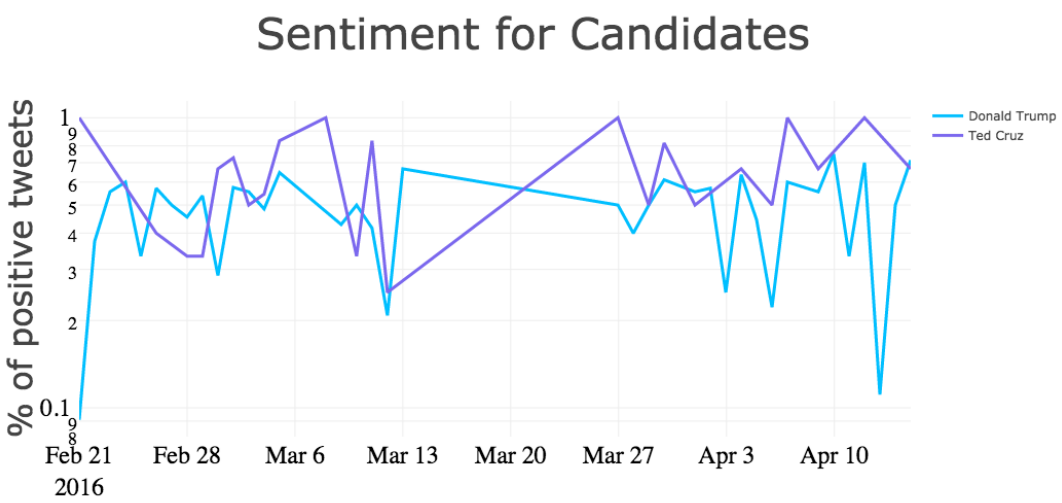

Figure 8: Sentiment towards Republican candidates in New York

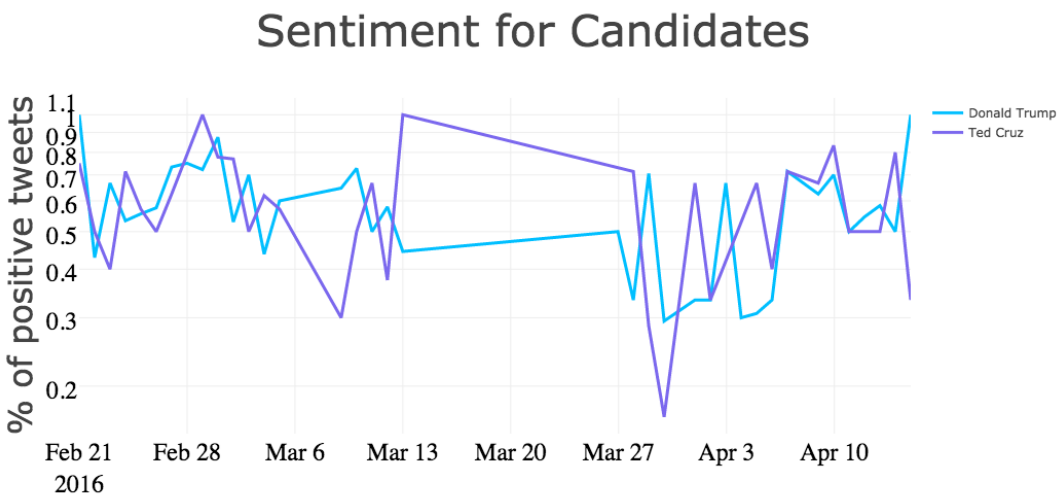

Figure 9: Sentiment towards Republican candidates in Texas

candidates. Fig.10 and Fig.11 shows the public sentiment towards Hillary Clinton and Bernie Sanders in New York and Texas respectively. From the plots it is clear that there is no much differentiation between both these candidates in these states.

4.4 Election predictor

From the observed data, it is clear that twitter data is not a good tool to predict election results. Donald Trump despite of a very low positive sentiment has had a very successful campaign while Bernie Sanders has had a relatively less successful campaign as compared to Hillary Clinton even though Sanders had clearly more positive sentiment on twitter. The sentiment is New York was in favor of Ted Cruz but Donald Trump won the primary in New York by a huge margin (60% votes to Trump as opposed to 15% to Cruz). Similarly, Donald Trump had almost similar sentiment score as Ted Cruz in Texas but Cruz ended up as a winner with 43% of the votes as opposed to 26% for Trump. Therefore, we conclude that Twitter is not a great tool for predicting election results instead it is a good tool for campaign managers to see how people react to various statements and debates. Also, it is important to recognize that not all of the voters are active on social media and therefore these trends may not be representative of the voting population.

5. CONCLUSION AND FUTURE WORK

We will discuss what we have learned and potential future work in this section.

5.1 Lesson Learned

In conclusion, we think we have learned a couple of things from this project.

We first learned how to utilize Spark framework to process data. Spark, arguably the hottest rising star in computation paradigm, is really easy to learn from scratch. It has well-documented tutorial that walk you through how to

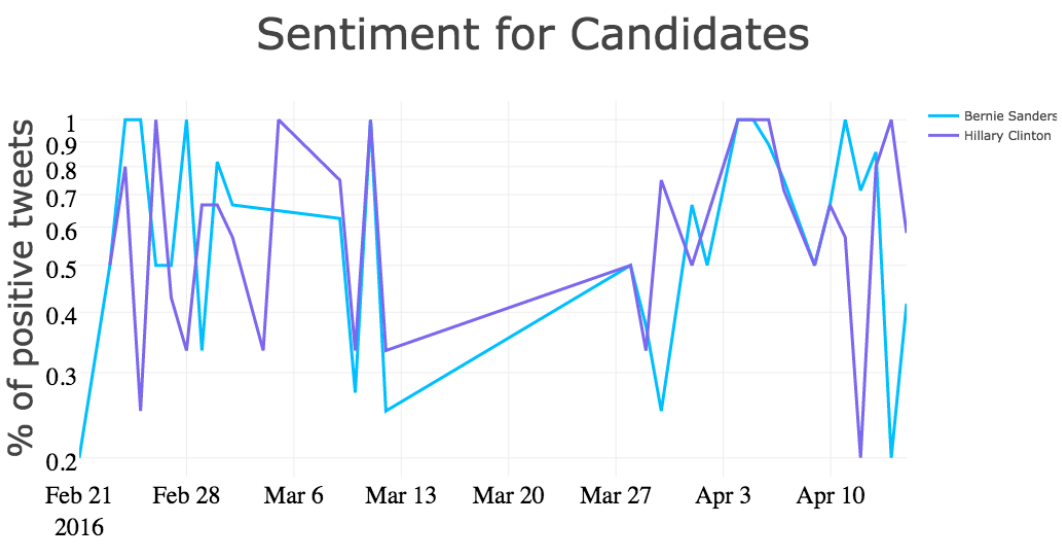

Figure 10: Sentiment towards Democratic candidates in New York

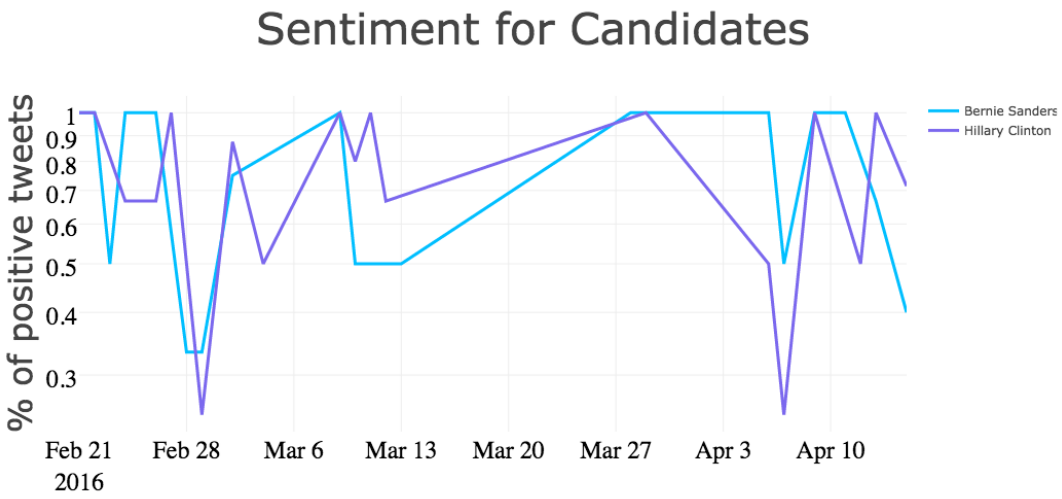

Figure 11: Sentiment towards Democratic candidates in Texas

setup components and how to connect different components. More importantly, it is way faster than most other solutions out there for our purpose. For similar project, we highly recommend using Spark over other framework.

Secondly, we also learned more about sentimental analysis and classifiers. As in the lecture, we mainly focus on text mining and information retrieval, yet only talk a little about sentimental analysis and classifiers. Specifically, we only talked about logistic regression and latent aspect rating analysis for sentimental analysis[9], and only touch the surface of SVM, Naive Bayes. After additional studies outside the scope of the class, we found a variety of usages of those classifiers. For example, Naive Bayes classifier is a good approach to our problem of analyzing sentiment for tweets. In summary, not only we know how to leverage those techniques in real projects, but we also learned how these techniques are used and implemented and what are their use cases.

Last but not least, we learned an important lesson that the volume of the data will impact the design process a lot. For this project, we took advantage of over 270 GB data, which is almost impossible to store on a single machine. It is fair to say that the majority of our design is to serve for this huge chunk of data.

5.2 Future work

We think there are two aspects that we can enhance our system.

- **Using sources other than Twitter**: It is hard to deny that our project is still biased towards Twitter users, many of which are young people. This fact puts constraint on our problem definition. To get rid of such constraint, we need more reliable data source. A potential improvement to our project would be taking

in opinion from other sources.

- **Using multiple sentiment analysis tools**: For almost the same reason, our sentimental analysis component is trained on given text corpus. The problem of this approach is that it is biased towards the training set as well. One improvement we can make is to collect labeled Twitter data.

5.3 Special Thanks

Speical Thanks to Professor Chengxiang Zhai for providing the data we needed for this project!

6. REFERENCES

[1] Apache hadoop, http://hadoop.apache.org/.
[2] Apache spark, http://spark.apache.org/.
[3] Apache storm, http://storm.apache.org/.
[4] Historical presidential election information, http://www.270towin.com/states/.
[5] Natural language toolkit, http://www.nltk.org/.
[6] Plotly, https://plot.ly/.
[7] S. Bennett. Twitter usa: 48.2 million users now, http://www.adweek.com/socialtimes/twitter-usa-users/496836.
[8] C. J. Hutto and E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International AAAI Conference on Weblogs and Social Media*, 2014.
[9] H. Wang, Y. Lu, and C. Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 783–792. ACM, 2010.