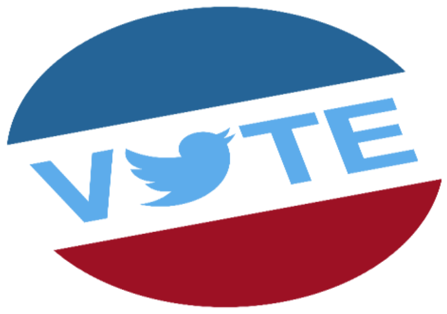
INTERNATIONAL
HELLENIC
UNIVERSITY

Predicting the USA presidential elections using Twitter Data

Student: Lazaros Oikonomou

SID: 3301140017



SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Information and Communication Systems

DECEMBER 2016

THESSALONIKI – GREECE

INTERNATIONAL
HELLENIC
UNIVERSITY

Predicting the USA presidential elections using Twitter Data

Student: Lazaros Oikonomou
SID: 3301140017

Supervisor:	Assist. Prof. C. Tjortjis
Supervising Committee Members:	Dr Christos Berberidis
	Prof. G. Evangelidis

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Information and Communication Systems

DECEMBER 2016
THESSALONIKI – GREECE

Abstract

This Dissertation was written for the International Hellenic University's MSc in Information and Communication Technology systems by student Lazaros Oikonomou under the supervision of Dr Christos Tjortjis.

Social media is a part of our lives for some years now. More and more people tend to express themselves via social media. This is an opportunity for analysts to exploit the information available for free in these websites, and use them to foresee future results for events like elections, product releases and many more. In this dissertation we are going to gather information from the famous social media Twitter and analyze it with the purpose of predicting the results of the upcoming elections in the United States of America. We will use different tools and develop code that enables and enhances the analyzing process.

I would like to thank my supervisor Dr Christos Tjortjis for his patience and for giving me the opportunity to work with him on a very challenging and also extremely interesting project. The support I got from him was exactly what I needed to finish this project successfully.

I would also like to thank my father and mother for supporting me throughout this process and my cousin Thodoris Amanatidis for giving me some extremely useful pieces of technical advice.

Lazaros Oikonomou
23/12/2016

LIST OF FIGURES

1. Introduction

In the first chapter of this dissertation we will try to provide a high-level analysis of the problem we are trying to address. We will emphasize the importance of Twitter as a tool for data analytics and examine how we can draw useful conclusions for important events (like elections etc.) by processing data from the popular social network. Furthermore, we will explain in detail what our objectives are and we will provide useful information about the USA elections that we try to predict. Finally, we are going to describe what the structure of the project will be and proceed with the implementation chapters.

1.1 The problem

Nowadays, more and more people choose to express themselves via the Internet and more specifically the social media. While these lines are written, over two billion users have access to social media accounts in websites like Facebook, Twitter, Instagram, Snapchat and many more, expressing their opinion, feelings, uploading photos, videos and status updates daily. As a result, large amounts of data are created every second, making social media a valuable source for data analysts all over the globe. If this data is analyzed properly, analysts will have the opportunity to gather information and draw useful conclusions for a variety of topics. However, there are a lot of issues that we will have to deal with during this task. In this project we will try to gather as many data as possible and analyze it with the purpose to make assumptions and predictions for the upcoming elections in America. Thus, the target of this project has two basic dimensions: Find the optimal way to gather tweets that are about the elections and try to see the sentiment they express towards the two candidates in order to make an accurate prediction about the elections.

The first issue we have to deal with when talking about social media and their data, is how to retrieve the data we want. We don’t want random data. We want data that refer to the elections. At this point we have to point out that the data we will use in this project is textual data. If we want to gather other formats of data the task becomes even more challenging and complex than it already is.

Twitter provides an API that is built for developers and analysts who want to fetch tweets and use them to perform data analysis. The challenge here is to make the right decisions and exploit the Twitter’s API capabilities in the best way possible. There are a lot of different choices that Twitter offers and we will make sure to use as many features as we can in order to make the data gathering process as sophisticated and rich as possible.

After acquiring the data we want, the preprocessing part comes next. We will describe how we altered our data without jeopardizing their integrity in order to make the analysis process easier. Data preprocess can be a tricky task and the decisions we make while doing it are extremely important for what follows next.

After the data gathering and preprocessing parts are over the most crucial part follows. We will try to build a system that will analyze the textual data we have and extract information by revealing the sentiment behind every tweet we gather. This process is called sentiment analysis (or opinion mining) and we will talk in detail about it in the following chapters. There are a lot of different ways to perform sentiment analysis as we will see later and thus, we will have to make sure that the decisions we make for our methodology are based on practical criteria.

Since our mission is to gather information about the USA elections we had to make a decision regarding which social media is considered to be the most politically relative. After a research we conducted we came to the conclusion that the social media we were looking for was Twitter. Let's see how we made this decision.

1.2 Why Twitter?

Twitter is a social network with 313.000.000 active users at the time. It is also considered one of the richest media in terms of information volume and it is also considered the most extensive microblogging service on the planet right now. The difference with other social media like Facebook and Instagram is that Twitter is not based on images and videos but mostly in textual context. People write tweets to express themselves and make statements about different issues that might concern them. Consequently, if someone wants to express an opinion about a serious topic it is most likely to do it on Twitter rather than other social media.

In addition, as we mentioned earlier, Twitter offers API services for developers that make it easier to gather data. Despite the fact that Twitter is very careful regarding privacy and security, it offers the possibility to download tweets by making simple and easy requests to the API. The queries that will retrieve data can be customized as we will see later and the possibilities that we have when making requests are unlimited. We will elaborate on the data gathering process on the following chapters. Now let's see how exactly users use Twitter in order to express their opinion.

Every user that has an active account on Twitter can post messages up to 140 characters, called tweets. These messages may refer to a variety of topics. Actually, the user can write whatever he/she wants about any topic. The topic that is referred in a tweet is usually presented with a hashtag. A hashtag is a word that the users include in their tweets in order to make them more specific. For example in our case, tweets that are about the elections of USA that will occur in November 8th 2016 can be addressed with hashtags like #Elections2016, #8Nov, #Republicans, #Democrats etc. If a tweet contains hashtags like the ones we mentioned, it is most likely to be an opinion about the elections. Hashtags are very important for our project because we will use them in order to get data that are filtered and are about a specific topic (the USA presidential elections).

1.3 Dissertation’s objectives

This dissertation has two main objectives.

1. The first one is to use PHP in order to collect data from the Twitter API. Most researchers that collect data from social media chose different programming languages to do this task. In most cases the data collection and the data analysis are implemented using the same programming language. In our case, in order to do a more flexible and thorough analysis we chose to collect data with PHP. PHP is the best solution for project that include communication with APIs. The data gathering process has specific subtasks that need to be through before proceeding with data analysis. Those subtasks are:

   a. Build an application that will gather data from Twitter. In order to gain access to the APIs that Twitter offers, we first need to build an application. This is a typical procedure that reassures Twitter that we are actual humans and not malicious systems that try to jeopardize Twitter’s integrity. After building this application we get the credentials we need in order to make requests to the API and retrieve the data we want.
   b. Data gathering must be done with caution. The credibility of the data is one of the most important aspects of this task. To achieve this we will write a PHP code that avoids specific malfunctions like data repetition.
   c. Find a way to exploit all the abilities that the Search API of Twitter offers in order to collect as many data as possible. The more data we gather and analyze the more accurate our final predictions will be.
   d. Preprocess the data before starting to analyze it. Minimize the information that each tweet contains and keep only important details to perform sentiment analysis. Tweets have a lot of information when we collect them. When we make a request for tweets with the Search API the return results contain unnecessary information like usernames, time, hashtags etc. We will try to avoid getting useless metadata and keep as few as possible.
   e. Develop code that collects data that fulfill specific queries. As we will see later on we chose specific features for every tweet we gather. We wanted to collect tweets that contain specific hashtags and come from specific geographical locations.
   f. Develop code that will be able to handle requests for vast volumes of data. When the volume of data is too big there might be cases that the script crashes. We have to develop code that deals with crash issues and it’s not compromised even when the data retrieved is too much.

2. The second objective of this dissertation is to perform sentiment analysis (or as some call it opinion mining) on the data we gathered in order to be able to make a prediction about the USA 2016 presidential elections. The code is written in Python. Now in order to develop a code that does the task we mentioned we have to complete some subtasks as well.

a. We have to decide between different ways of performing sentiment analysis. We decided to use classification.
b. In order to perform classification we need to train a classifier that will be based on training data
c. The training data we will use, has to be relative to our subject in order to be as accurate as possible.
d. Find the appropriate libraries and methods that will help us classify the tweets we gather as positive, negative and neutral.
e. Make the code as simple as possible so we can use it anytime on different data and do the analysis process with credibility and efficiency.

Of course these two objectives serve a higher purpose for this dissertation which is to check if Twitter is a trustworthy social media to draw conclusions about serious topics like elections. If the results are accurate and satisfying we will be able to use Twitter as a tool in order to see what the public opinion indicates about various topics.

1.4 About the 2016 USA elections

On Tuesday November 8th, the United States of America, the most powerful nation on earth elects its new leader. After one of the most expensive campaigns in history, either the Democrats or the Republicans will have to form a Government that controls the largest army on the planet. It is easily understood that the 2016 USA elections are considered to be one of the most important events of our time. The result of the elections affect all of us and it’s very important to be well-informed before we start our implementation process.

1.4.1 The candidates

There are two political parties that are considered dominant in the United States and fight for the win. The Democrats and the Republicans. Before the elections’ campaign started these two parties had to choose who will represent them as the main candidate for the presidential elections. Now in order to pick who the presidential nominee will be for each party, they conduct their own elections beforehand. That way the candidate that will represent the party is chosen from the party’s delegates. When we are referring to a party’s delegates we are talking about voters with opinions that matter more than an average voter within the party. The most used term to describe those voters is party grandees that have power and are considered as important individuals that get to decide things in the party’s conventions. The number of delegates on each state is determined by the state’s size and population.

After the candidates for each party are determined, the presidential campaign starts. The two candidates for the elections on November 8th:

- For the Democrat party, the former United States Secretary of The New York State, Hillary Clinton along with the former United States Senator Tim Kaine as vice president candidate.
- For the Republican party, the well-known businessman Donald J. Trump along with the former Indiana Governor Mike Pence as vice president candidate.

Of course there are other candidates as well but the probability of a different winner is almost zero. For the record we will refer to some candidates that are considered important as well. There is the Libertarian party with former Governor Gary Johnson as presidential candidate, the Green party Dr. Jill Stein from Massachusetts and the Constitution party of the United States with Darrell Castle from Tennessee as presidential candidate.

In our project we focus on the two most popular candidates, Hillary Clinton and Donald J. Trump. Both candidates are controversial personalities with a lot of stories behind them.

Donald J. Trump's campaign was debatable from the beginning. He stated that he considers Mexican immigrants as "rapists and criminals" and he talked about building a wall to keep illegal immigrants from entering the United States. He also started conflicts with a US judge, an ex-Miss Universe, A Fox news anchor and Muslim society in general after some statements he made. He is also involved in a tax scandal as he refuses to publish his tax returns and also avoids to pay his taxes for the past 18 years.

On the other hand, Hillary Clinton is not a well-accepted public persona either. The email scandal is known worldwide and damaged her image. Also a lot of questions are risen about the donations that have been made to the Clinton Foundation.

We can easily understand that the two candidates have a lot of negative publicity on their heads. What we are going to do is see how Twitter users react in these kinds of situations. We are going to perform a sentiment analysis that will classify Twitter users as Trump or Hillary supporters depending on what they choose to post on Twitter. Of course it's not simple to collect data from all over the country. We wanted to make our project more targeted. Consequently, we decided to focus on the most important states that are more likely to determine who will be the next President of the country. Let's see which states are considered the most irresolute.

1.4.2 The Key states

Since we don't have the equipment to retrieve data from all the States, we conducted a thorough research in order to see which states are the most important and should determine the final results on November 8.

A term we need to make clear to the readers before continuing is the electoral votes of each state. The system we are going to describe was established in 1845 and is called the Electoral College. This is actually the body of individuals that will be the most decisive in the elections and will elect the President and Vice-President of the US. Most of us believe that the President is elected with popular vote, but this is far from true. The people that actually determine the results of elections are called electors. Each state has a different number of electors depending on how many congress members the State has. There are 538 electors in total. Below we can see the electoral map of the United States of America taken from http://www.politico.com/2016-election/results/map/president.

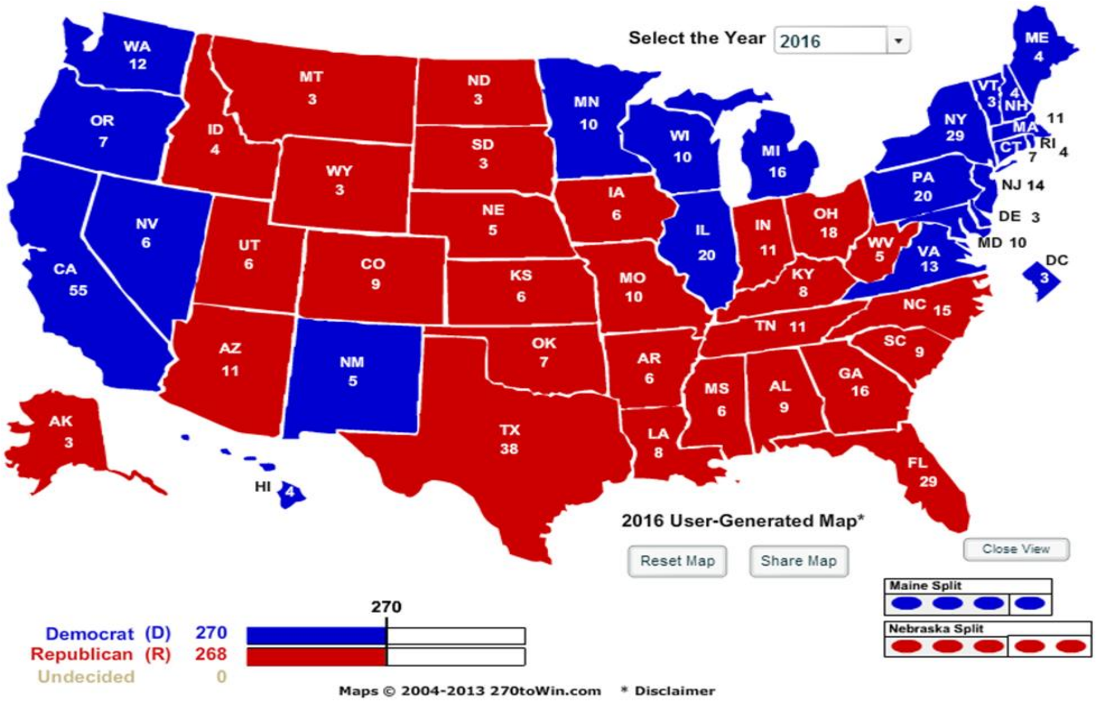


Figure 1: *electoral map of USA [1]*

The state with the most electoral votes is California with 55. From the states that the results are not clear on who is going to win yet, the one with the most electoral votes is Florida with 29. Despite the fact that there are states with more electoral votes, Florida seems to be divided in to two equal shares regarding the election results. This is why it is considered to be one of the most important States in the elections.

More States that are considered crucial for the final results are Ohio and Pennsylvania. Past statistics indicate that whoever wins two out of the three states we mentioned is going to be the final winner. However in the elections of 2016 things seem to be a little bit different. Now the most important states that there are no accurate polls yet are:

- Iowa, with 6 electoral votes
- Colorado, with 9 electoral votes
- Minnesota, with 6 electoral votes
- Arizona, with 11 electoral votes and finally
- North Carolina, with 15 electoral votes

Now in order to make our data sample accurate we will have to choose to fetch tweets from three states. The states that we decided to collect data from are the following:

- Florida
- Ohio
- N. Carolina

These are the three states will determine the final results. Now a very important aspect of our project is the time we collect our data. There are some important dates that the candidates will be more exposed to public opinion and that's when we have to focus on in order to get our data.

1.4.3 The Important dates

There are some dates that are considered crucial for the upcoming elections on November. For this project it is crucial to gather our data in the right time when people have seen discussions, debates, reactions etc. in order to make their decision for their vote. The most important dates for the elections are the following:

- On September 26th we have the first Presidential debate between Donald J. Trump and Hillary Clinton
- On October 4th we have the first Vice Presidential debate between the two candidates for Vice President
- On October 9th we have the second Presidential debate
- On October 19th we have the last Presidential debate
- On November 8th we have the Presidential Elections

Our intention is to retrieve tweets after these dates and also the final days before the elections in order to see how the voters react as the elections are coming closer.

1.5 Structure

After the introduction we continue with the Literature Review chapter (chapter 2). In this section we will take a look in similar projects, their methodology and we will briefly describe how our work will be different. There are a lot of projects that have similar objectives as we will see, but we make it clear that this project will differentiate from others and do the whole process with a different point of view.

Chapter 3 is the Data Gathering chapter and it contains a detailed description of the implementation process for retrieving data. It also describes how the Twitter APIs work and what we have to do in order to use them to gather the data we want. There are a lot of issues when we make requests with APIs and in this chapter we will describe how we will deal with problems that might occur while collecting tweets. In addition, we provide a detailed description of the PHP code we built, explaining every parameter we used and every decision we made while collecting our data.

Chapter 4 is called Sentiment Analysis and it thoroughly describes the decisions we made regarding data preprocess and data analysis, what methods we chose to include and how we used them in order to classify tweets as positive, negative and neutral. Furthermore, we provide a detailed description of the Python code we wrote.

Chapter 5 is called Testing and Evaluation. Here we check if the code we wrote (both PHP and Python) work properly and we demonstrate some test cases to show that the implementation was successful. In addition we cite the actual results of the elections and compare them to ours in order to evaluate our work.

Chapter 6 is called Conclusions and future work. Here we describe the conclusions we draw while working on this project and make specific suggestions to future researchers that want to enhance this project on how to do it and which tools will be the most suitable to help them do it.

Chapter 7 is called References and Bibliography and it is where we cite the references we made while describing our work and the Bibliography we read before starting this project.

2 Literature Review

In this chapter we present the most relevant papers that are addressing the same problem. Also, we briefly mention some papers that have some interesting ideas about the subject and might be useful during this dissertation. We analyze the approaches behind those papers and try to give a description of their architecture. The papers we are going to examine are the following:

1. Tweets mining for French Presidential Election [2]
2. Mining Twitter Big Data to Predict 2013 Pakistan Election Winner [3]
3. Twitter as a tool for predicting election results [4]
4. Prediction of Election Result by Enhanced Sentiment Analysis on Twitter Data using Word Sense Disambiguation [5]
5. Emotion analysis of Twitter using Opinion mining [6]
6. Opinion Mining about a Product by Analyzing Public Tweets in Twitter [7]

2.1 Tweets mining for French Presidential Election [2]

In 2012 before the French presidential elections, Katarzyna Wegrzyn – Wolska and Lamine Bouguera published a paper called “tweets mining for French presidential election”. Their goal was to gather Twitter’s data (tweets) and transform them into useful information unveiling the voters’ preferences towards the upcoming elections.

There are two terms that are considered as key aspects of this paper: Text mining (TM) and Social Network Analysis (SNA). It is important to understand that the main challenge was not only to gather and process data but also to find patterns that link information together like a puzzle and help political analysts draw conclusions. There are a lot of difficulties that make this task challenging. Twitter is known to provide freedom of speech to its users, giving them the opportunity to express themselves in every way they desire. This might be the case in Democracy, but in this experiment, individuality in terms of speech and language might cause confusion. Difficulties like slang use, idioms and abbreviations that voters might choose to express themselves make the whole process more complicated than it already is.

Methodology

As we mentioned earlier the goal of this paper was to gather tweets and develop an application that will be able to perform content analysis by extracting knowledge and predicting trends. The premise was to gather the tweets that refer to presidential candidates and classify them as positive, negative and neutral. Twitter is evolving every day and users might become the help that data analysts need to connect the dots and come up with innovative ways to predict future events.

The system that was developed in this paper measures the frequency that a candidate is referred on tweets and also how the crowd reacts to their everyday public exposure (interviews, debates etc.).

The architecture of the system is based on the following: search and fetch tweets, analyze and classify them as negative, positive or neutral and store them in a well-formed database. To get the tweets needed for the project, the researchers used the most common and reliable way, which is the Twitter's REST API. After getting the tweets the pre-process phase involved deleting words that weren't useful (propositions, articles etc.). To extract knowledge from a tweet they had to keep only the most important words like adjectives and adverbs (words that reveal opinion), so that they are not distracted by unnecessary information. To achieve that, the writers decided to keep the following: For the candidates, the Id, Name and Id of the tweet that the candidate was mentioned in. For the notice, Id, type, value and Id of the candidate that the tweet is about. The system also includes a user's interface that is a search platform that gives the opportunity to make individual dictionaries and determine which words will be considered as positive or negative.

The system's architecture is shown below.

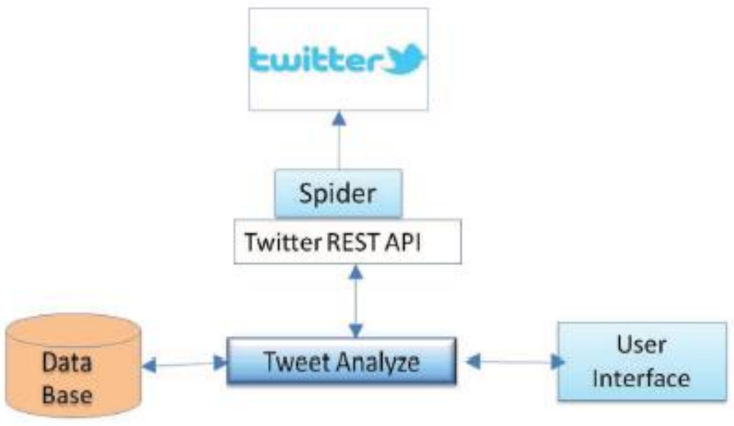


Figure 2: *system's architecture [2]*

A very important part of the architecture is Sentiment analysis. To demonstrate how difficult the sentiment analysis process is, we are going to see an example of a tweet that the system has to classify as positive, negative or neutral. Here is the tweet: "Yeah, for me the best candidate. I have only met 2 people in real life and 1 person on the net who hates this program. My favorite one ever!" This is a tweet from a potential voter. The polarity of the tweet is what has to be determined. The tweet is divided and analyzed one word at a time. Even though it is easily concluded that this is a positive tweet, there are phrases that might cause confusion. For example the phrase "…who hates this program" has a negative polarity in most cases. However, in this case the voter is using this phrase to emphasize and not to criticize. This is a typical occasion of misleading text. Opinion can sometimes be expressed in different ways depending on individuals. For example we might have a tweet that involves sarcasm which makes it extremely difficult for the system to understand and define if it is a positive or a negative reference. Furthermore, the vocabulary that users might choose to express their opinion might be misleading as well. Slang terms and idioms is a challenge for the system to understand and in most cases it is very confusing.

Another challenge introduced in this paper is the classification between objective and subjective phrases within the tweets. This is important because the objective phrases do not need to be processed. On the other hand, the subjective phrases are the ones that need to be analyzed in order to understand what the voter is saying about the candidates. There are many approaches that can be adopted to resolve this issue. Adjectives are considered to be the most important words because they are the ones that show sentiment.

Semantic orientation is a term that the writers consider extremely important. This term refers to words' polarity (negative or positive) and also has an intensity level of sentiment. This means that a phrase can be "less negative" or "more positive" than other phrases that fall under the same category. The semantic orientation (SO) minus A, where A (word, pword) is the correlation between the word and the equivalent negative word. So:

$$SO - A(word) = \Sigma_{p \in P} A(word, p) - \Sigma_{n \in N} A(word, n)$$

We can see that the semantic orientation is measured as the total sum of positive words minus the total sum of negative words. If the result is a positive number, then the phrase is considered to be a positive reference. If the result is a negative number the phrase is considered to be a negative reference. The absolute value of the result indicates the intensity of either positive or negative outcome. Now, if we want to find the association between word – A, there are different approaches that might be helpful. The most commonly used approach is by Church and Hanks, it is called Pointwise Mutual Information (PMI) and it's calculated like this:

$$PMI(mot_1, mot_2) = log_2 \frac{p(word_1 \& word_2)}{p(word_1)p(word_1)}$$

Where p ($word_x$&$word_y$) is the probability of presence of both words within the same phrase.

This paper introduces also some different ways to do sentiment analysis. One of them is Turney's approach that consists of four (4) different steps:

- Process every word of the phrase separately
- Place adverbs and adjectives together in mini two-word phrases
- Calculate semantic orientation minus Pointwise Mutual Information (SO - PMI) for each phrase that is generated by adjectives and adverbs
- Determine if the orientation is positive or negative

Another interesting approach for doing sentiment analysis is Pung's approach. This approach is based in processing subjective parts of text using a statistical classifier. It doesn’t separate phrases in segments, but it is based in phrase continuity that can help detect trends and preferences within a text.

The classification process of tweets as negative and positive in this paper is based on Pung’s approach. The candidates’ popularity is measured according to how many positive tweets each one receives. The following table shows the trends of voters among different time periods.

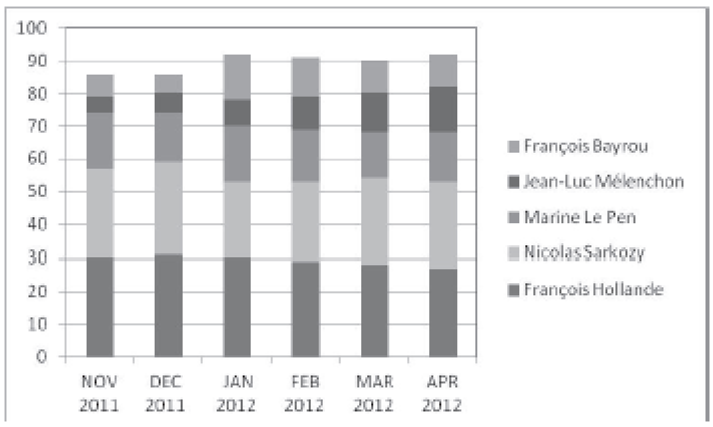


Figure 3: *trend of voters among different time periods [2]*

Conclusions

The purpose of this paper was not only to predict the election result, but also to analyze how different trends influence the mass in social media using different polling methods. Furthermore, the meaning of opinion intensity is introduced giving the opportunity to draw conclusions about how loyal and determined voters are. As writers claim, social network analysis and text mining for political purposes is a field that has lot of challenges and it might become a useful and accurate method of predicting both political and economic trends in the future.

In our project we are going to be more specific about the classification process and make the data gathering more targeted in terms of geography in order to get more accurate results.

2.2 Mining Twitter Big Data to Predict 2013 Pakistan Election Winner [3]

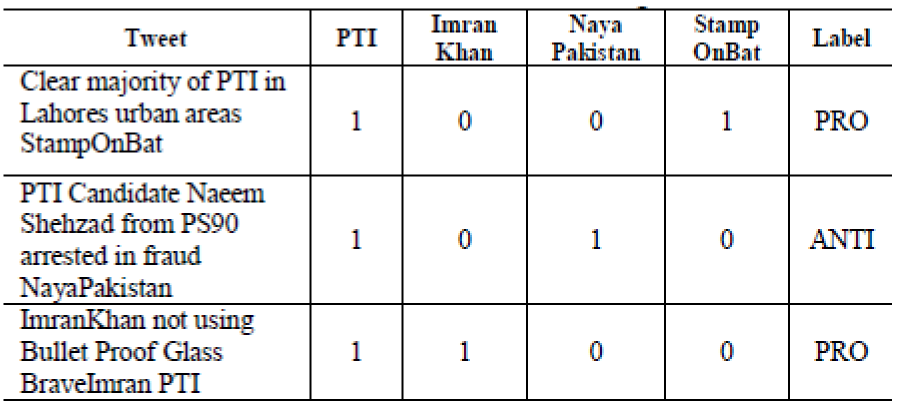
In 2013 Dr. Tariq Mahmood, Tasmiyah Iqbal, Faranz Amin, Wajeeta Lohanna and Atika Mustafa from the Department of Computer Science of the National University of Computer & Emerging Sciences in Karachi Pakistan, wrote a paper called “Mining Twitter Big Data to Predict 2013 Pakistan Election winner”. Their goal was to retrieve tweets, pre-process them, store them in a dataset and come up with ways to draw conclusions regarding the winner of the elections. Three parties were massively twitted at that time, Pakistan Tehreek-e-Insaaf (PTI), Pakistan Muslim League Nawaz (PMLN) and Muttahida Qaumi (MQM). PMLN was the party that prevailed at the end.

Methodology

According to the writers, a very important challenge for them was the data pre-process part. In order to be able to analyze tweets they needed to alter data and make them simpler to process, without jeopardizing integrity and authenticity of the dataset.

Tweets were gathered using a site/tool called *Twimemachine* that allows data scientists who want to retrieve tweets, to fetch them. More specifically, tweets during January 2013 to May 2013 were gathered. The elections were scheduled for the 11th of May. The tweets that were gathered were about the three parties mentioned earlier. Twenty-four users were monitored during this period. Fifteen out these users were political analysts and nine of them were normal every day users. The total amount of tweets gathered was 55.000, but only 9.000 were considered related to the elections.

As we mentioned earlier, the data that were fetched (9.000 tweets tin txt files) needed to be pre-processed in order to become simpler to analyze. This was achieved by: removing the usernames, the word “RT” (retweet), punctuation symbols, URLs etc. Each tweet that was about a political party was classified as positive (Pro) or negative (Anti). Tweets that were considered neutral by metrics, were not taken into account. Each party had a list of attributes. These attributes are mostly hashtags that refer to the party. If an attribute is in the tweet it takes the value “1”, if it is not in the tweets it takes the value “0”. Below we provide a table that was taken out of the paper to demonstrate how exactly this system works. The table is about tweets that refer to the party called “PTI”.



Tweet	PTI	Imran Khan	Naya Pakistan	Stamp OnBat	Label
Clear majority of PTI in Lahores urban areas StampOnBat	1	0	0	1	PRO
PTI Candidate Naeem Shehzad from PS90 arrested in fraud NayaPakistan	1	0	1	0	ANTI
ImranKhan not using Bullet Proof Glass BraveImran PTI	1	1	0	0	PRO

Figure 4: *tweets that refer to the PTI party [3]*

The predictive models that were used in this case are:

CHAID decision tree
Naïve Bayes
Support Vector Machine (SVM).

Rapid Miner was the tool that was used for tests with the three chosen predictive models. Tweets between 8th and 11th of May were processed in test metrics. The label field was left empty for Rapid Miner to predict.

The results of the whole process were very interesting. All three algorithms predicted pretty much the same results. However, the CHAID algorithm proved to be the most accurate. It also provides prediction rules that the other algorithms don't, so it was chosen to be the main algorithm of the project.

The party that was more positively commented in the tweets was PTI. The paper predicted that it would be the winner of the elections. However, the actual results were different. The party that won was PMLN. Second in votes was PPP, a party that was not taken under consideration by the writers. PTI was third but it was the victor in the Khyber Pakhtunkhaa region that is considered very important in Pakistan and also won more seats in the parliament that previous years.

Conclusions

The conclusion that is drawn from the paper is that there was in fact a twitter-based trend towards PTI that helped the party gain more votes than it was originally meant to get. Even though the final results were different, we can say that Twitter and social media in general can be used to predict the voters' trends and draw conclusions about elections.

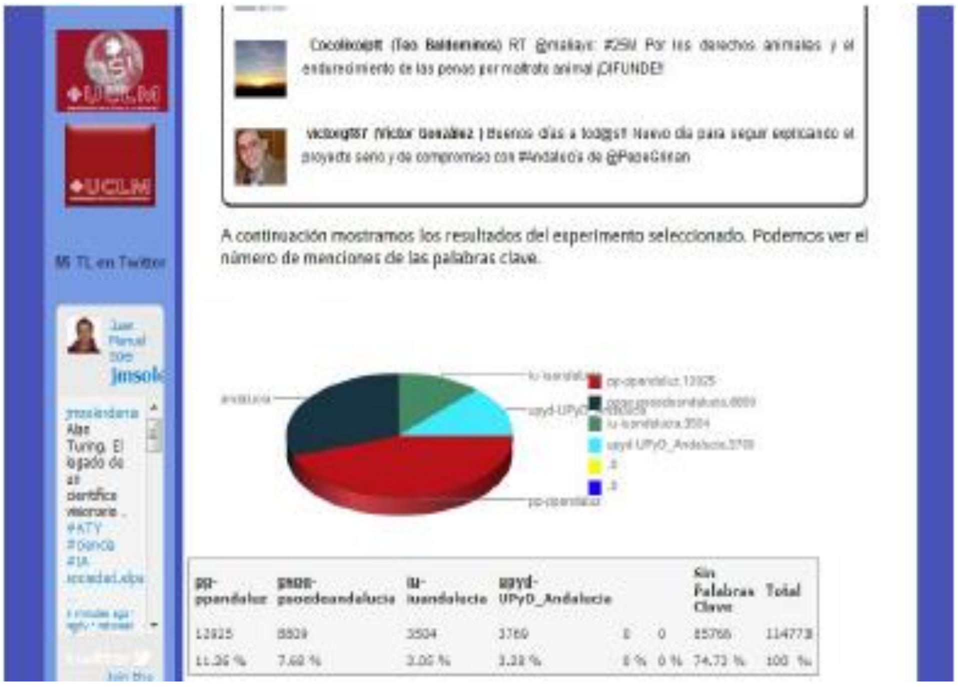
In our project, the data gathering process is done differently with the help of the Twitter's Search API and also the tweets are not gathered from specific individuals but from random citizens of USA.

2.3 Twitter as a tool for predicting election results [4]

Juan M. Soler , Fernando Cuartero and Manuel Roblizo from the Instituto de Inform´atica de Albacete, published a paper in 2012 called "Twitter as a tool for predicting election results". The whole premise was to get tweets that refer to 2011 and 2012 elections in Spain and try to process them in order to predict the results. To understand how important social media were for Spain at that time the writers indicate that 95.7% of Spanish internet users were using social networks like Twitter, Facebook etc. Furthermore, it is highlighted that after the Obama campaign and the vast use of internet and social media from his public relation team, more and more political parties invest huge amounts of money in advertising via social media. That makes analyzing and processing of tweets even more important for political analysts and data scientists.

Methodology

The tool that is introduced in this paper is *TaraTweet.* This tool has a two-fold purpose. On one hand, TaraTweet can help data analysts observe Twitter's conversations that involve specific hashtags. Users are the ones that will define what hashtags they want to retrieve and filter them depending on what their goal is. This is important because the users can customize the search and adjust it depending on what they want to achieve. On the other hand, this tool has the ability to count keywords in order to predict trends. The keywords are chosen by the users as well. The TaraTweet tool is displayed below.


[illegible] (Teo Baldeminos) RT @[illegible]: #25M Por los derechos animales y el endurecimiento de las penas por maltrato animal ¡DIFUNDE!

[illegible] (Víctor González) | Buenos días a tod@s!! Nuevo día para seguir explicando el proyecto serio y de compromiso con #Andalucía de @PepeGriñan

A continuación mostramos los resultados del experimento seleccionado. Podemos ver el número de menciones de las palabras clave.

pp-ppandaluz	psoe-psoedeandalucia	iu-iuandalucia	upyd-UPyD_Andalucia			Sin Palabras Clave	Total
12925	8809	3504	3769	0	0	85766	114773
11.26 %	7.68 %	3.05 %	3.28 %	0 %	0 %	74.73 %	100 %

Figure 5: *TaraTweet's interface [4]*

When an experiment is conducted in TaraTweet, the researchers that conduct the experiment are able to follow the hashtags that they chose and see how often they appear. Tweets are retrieved using the Twitter's API (just like what we saw in the first paper about French presidential elections). To demonstrate the results of each experiment online the writers chose the Google chart API.

Three major experiments were conducted. The first is about the Spanish regional elections on 22th of May 2011. The second experiment is based on general elections that occurred on November 20 2011. The third and final experiment was about the elections in Andalucía conducted also in 2011.

Regional Elections

The first experiment was about the regional elections in Spain in 2011. This attempt started on May the 5th and ended on June the 2nd. The hashtags that were taken under consideration were the following: "#22m", "#elecciones2011" and #elecciones. If a tweet included one of these hashtags it was considered related to the elections. During these 7 days of data gathering the researchers downloaded 102.282 tweets overall. At that point five parties were considered as dominant: "PP", "PSOE", "IU", "CIU" and "UPyD". From the first stages of the analyzing process the crowd showed a strong preference to PP.

Each word in tweets was counted separately. That means that if a tweet contained more than one reference, it would be counted more than once. One of the main challenges of this project was the string analysis of hashtags. When PP was mentioned most of the hashtags contained the string "PP". On the other hand, when a hashtag referenced PSOE, there was a variety of different string characters that made the process more complex. Strings like “PSC”, “PSCM”,”PSPV”,”PSE” were about PSOE and the system needed to be able to count these references as related data. The same thing happened with CIU.

After gathering the data, the tweets for each party were counted. Below we can see the mentions’ pie chart indicating how many times each party was referenced in tweets and the votes’ pie chart showing the actual results of the elections.

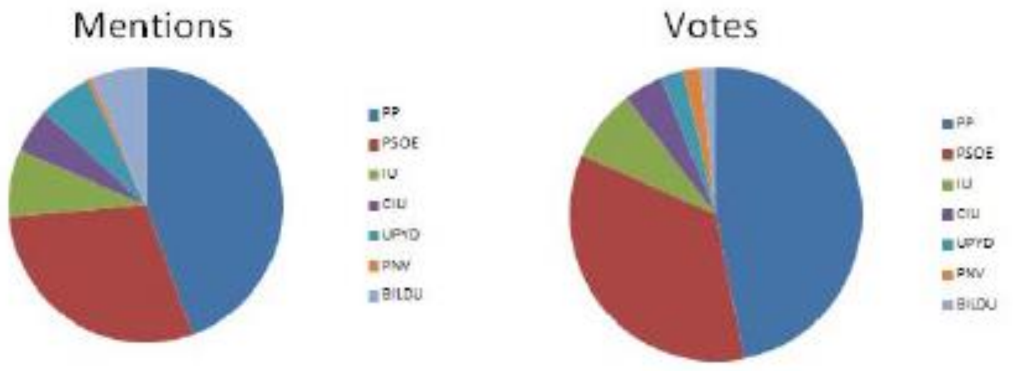


Figure 6: *mentions’ and votes pie chart [4]*

We can see how close the results of the two charts are. What we can easily understand is that two parties received more votes than the research predicted. That happened due to poor social media advertisement by the parties. The parties that weren’t advertised properly in social media were PSOE and PNV. We have to say that in the PNV case the results were influenced by the different names that the party was mentioned in tweets. Parties that were mentioned more regularly but performed worse in elections are UPyD and Bildu. We can say that they handled social media better giving the impression they were massively accepted by voters although they weren’t.

General elections

In this experiment, two hashtags were taken into account: “#20n” and “#elecciones”. The experiment started on August 4th and ended on December 2nd 2011. The goal was to observe what twitter users think about the upcoming elections and come up with predictions. The number of tweets that were gathered was 259.016. More tweets means better accuracy in the results. The political parties that were mentioned more frequently in tweets were: “PP”, “PSOE”, “IU”, “CIU”, “UPyD” and Bildu. The last one was added due to the unexpected results of the previous elections. As in the previous case, PP seemed to have an advantage towards the other parties. Below we can see the mentions of each party on Twitter and the actual votes it gathered.

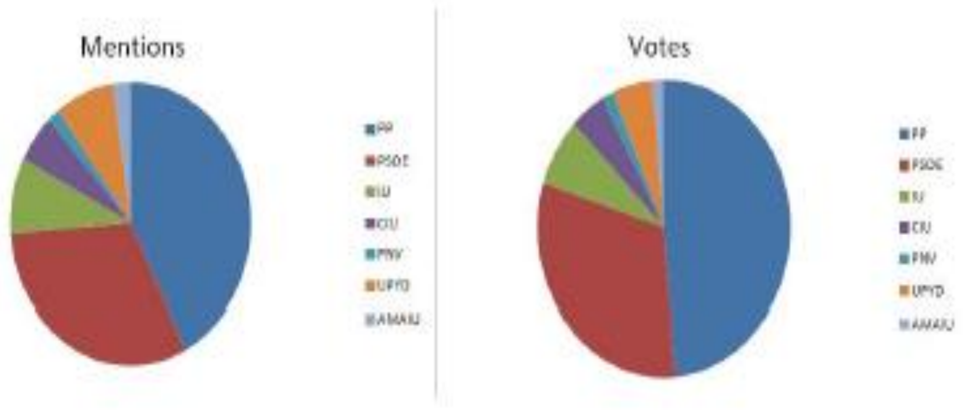


Figure 7: *mentions' and votes' pie chart [4]*

As in the case of the previous elections, we can say that the two charts are quiet similar to each other. The left one is about how many times a party was mentioned in tweets (regardless of positive or negative opinion) and the right one is the chart with the actual result. Again, there were parties that performed better than expected and parties that didn't go as well as the predictions indicated.

Due to the large amount of tweets processed, the risk of data manipulation by activists or professional social media analysts that each party had, was something that needed to be taken care of. Consequently, the writers decided to take some actions in order to make the data sample trustworthy. What they did was to find which users were more active and setting a limit to the tweets that were under consideration. Users that are more active from others and post comments about political parties regularly are probably members of the parties and the writers didn't want to take such users into account.

Elections in Andalucía

The final experiment is about the elections that took place in Andalucía on March the 25th. The hashtags that were examined were: "#25m" and "#andalucia". The experiment started on January 19 and finished on March 27. The total amount of tweets gathered was more than 176.000. The most popular parties that participated in the elections were PP, PSOE, IU and UPyD. Once again, PP appeared to have a significant advantage.

It is important to understand that all three experiments counted only the mentions of parties on Twitter. That means that even if a tweet had a negative opinion for the party it mentioned, it was still considered in favor of the party. That way, these experiments calculate each party popularity without taking into account the emotions behind tweets.

The results of the elections were proven to be similar to the predictions of the writers. Below we can see the two charts.

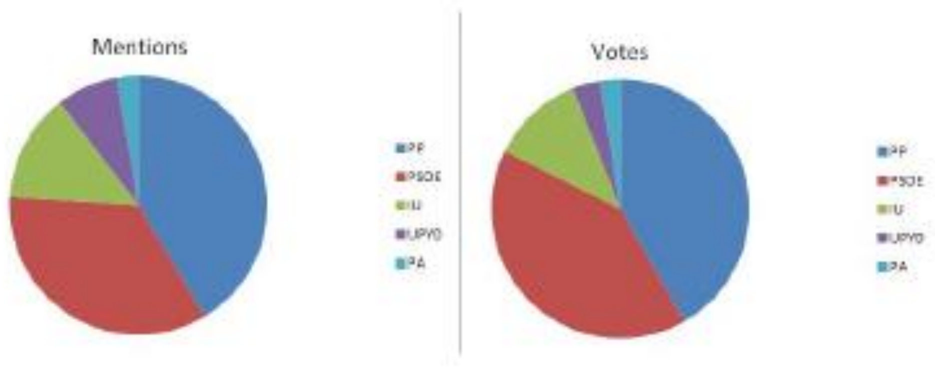


Figure 8: *mentions' and votes' pie chart [4]*

We can see that the percentage of the first party (PP) is accurately predicted. PSOE seemed to get less votes than it took because of bad social media promotion. IU is exactly the opposite of PSOE. It took less votes than it was originally predicted and that's probably because the promotion team of the party emphasized on social media. Same goes for UPyD. Here the difference is more obvious. The mention percentage was a lot higher than the votes' percentage.

Conclusions

The paper presents a new tool for Twitter analysis called *TaraTweet.* Using this tool 500.000 tweets were analyzed and the results were shown at the tool's site. The results of all three experiments conducted are very accurate. There are some parties that received more or less votes than originally predicted but this is normal since we are talking about elections. With these three experiments we can identify a correlation between mentions and actual vote intentions. Parties that invest in social media promotion are more likely to see satisfying results in the elections. This paper makes clear we can easily trust Twitter analysis to conduct experiments and come up with results that are actually really close to real-time vote intentions.

However, the methodology used by this paper is a bit controversial. Not doing sentiment analysis on tweets and just counting how many times a party is mentioned in not the best way to draw conclusions. This dissertation is going to dive into more detail and perform text and opinion mining as extensions to sentiment analysis. In addition, the data gathering process is done differently.

2.4 Prediction of Election Result by Enhanced Sentiment Analysis on Twitter Data using Word Sense Disambiguation [5]

Rincy Jose and Varghese S Chooralil from the Department of Computer Science and Engineering of Rajagiri School of Engineering and Technology published a paper in 2015 that proposed a new approach for sentiment analysis and sentiment classification. They used a novel method for analyzing tweets and with the help of lexical resources such as SentiWordNet, WordNet and Word Sense Disambiguation they tried to extract information and knowledge out of tweets. In addition, in order to achieve the highest accuracy possible they also proposed a negation handling approach in the pre-process phase.

The main goal of sentiment analysis is to determine the polarity of a single text. In most cases the results of sentiment analysis have three possible values: positive, negative and neutral. Three Sentiment analysis techniques are considered the most important in the field according to the writers. Those are:

- The machine learning approach
- The lexicon based approach
- The hybrid approach

The first one uses the basic linguistic features and some of the well-known machine learning algorithms. The second, is divided in two further techniques. The first one is called dictionary approach and the second is a corpus-based approach. The hybrid approach has techniques from both approaches. In this paper the writers chose to go with lexicon based approach using tools like WordNet and SentiWordNet to observe political statements from Twitter users.

Methodology

The application that was developed during this paper consists of three basic operations: Data acquisition, Pre-processing of tweets and Sentiment classification and analysis. In the phase of Data acquisition the writers chose the Twitter's streaming API that gives the possibility to retrieve real-time data. In the pre-process phase, they chose to delete unnecessary information that were in the tweets like the username, the hashtag symbol etc.

Negation handling was extremely useful in order to achieve high accuracy. Let's see what this means. In text mining there are some issues that need to be dealt with. For example, phrases like "it's good" and "it's not good" are very dangerous to end up having the same meaning if the system is not well-designed. In this case, the writers decided to use an algorithm for dealing with these kinds of issues using bootstrapping and state variables. In addition, some words that can be considered to be highly sentimental are only in their natural form in the dataset. For example the word "fail" is considered a highly sentimental word with a negative impact. The writers decided to add the opposite form of these kinds of words in the dataset. This means that when the word "fail" is found in a tweet, the tweet is added in the negative class and also the word "not_fail" is added in the positive class. This approach proved to be very effective because the accuracy was improved and the sample was simpler for analyzing.

The sentiment analysis and classification is done using WordNet and SentiWordNet. WordNet is a database of the English language that groups words based on their synonyms. The words are separated in verbs, adverbs, nouns, adjectives and so on. SentiWordNet is an extension of WordNet that classifies groups of words in three different classes by assigning them a score (depending on the words that each group contains). The three possible scores are:

- [0,1] ➔ positive sentiment score

- [0,1] ➔ negative sentiment score
- [0,1] ➔ neutral sentiment score

The closer the score is to 1 the more positive, negative or neutral the phrase is.

Of course the writers couldn't just rely on WordNet for their sentiment analysis. In some cases the result of the synonyms that the tool uses to extract sentiment, don't have the exact same meaning with the original words. To deal with this problem they used sense-tagged word lists. Let's take a look at the Word Sense Disambiguation which is based in the first two tools we described earlier. Firstly, a tokenization process and speech tagging was done for each tweet. Then, in order to classify tweets the Word Sense Disambiguation was applied. What this tool does is to find the best synonym phrase for each tweet within the WordNet database. Afterwards, the SentiWordNet assigns weights to the words of each phrase. By applying the three tools we mentioned every tweet is now classified as positive, neutral or negative.

The methodology that the writers chose to go with was applied in 12000 tweets that were about the Delhi elections and more specifically about the two candidates Arvind Kejriwal and Kiran Bedi. The figure below shows what the results of the sentiment analysis were.

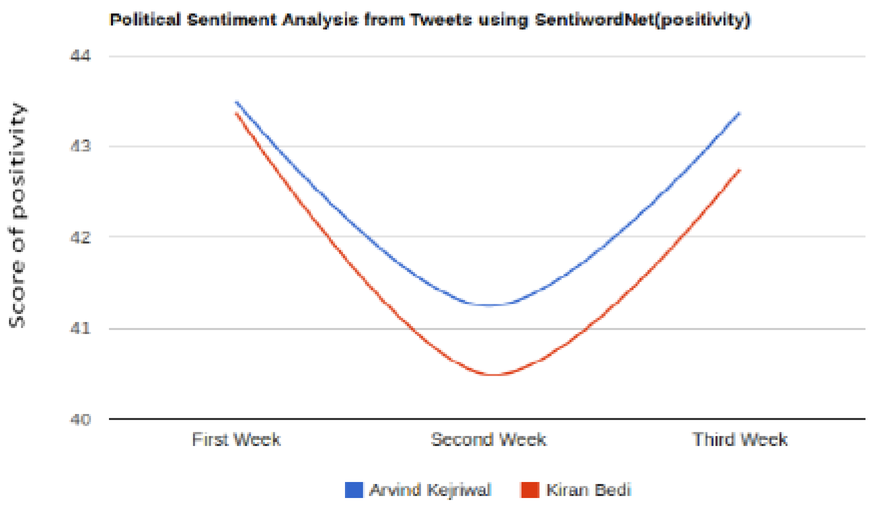


Figure 9: *results of sentiment analysis [5]*

It is clear that Arvind Kejriwal seems to be more positively tweeted by users. The final results of the elections are not shown in the paper.

Conclusions

Using the Twitter's streaming API the writers propose a sentiment analysis method for tweets that uses three very well-designed tools (WordNet, SentiWordNet and Word Sense Disambiguation). The writers' innovative spirit lead them to "play" with these tools. Finally, using these tools to achieve even higher accuracy and finding new ways to do sentiment analysis are some of the challenges that this paper introduces.

In our project we will use the Search API and not the Streaming API. We will also gather more data and not use lexicons, but Naïve Bayes classification model to do the sentiment analysis part.

2.5 Emotion analysis of Twitter using Opinion mining [6]

This paper is written by Akshi Kumar, Prakhar Dogra and Vikrant Dabas from the department of Computer Engineering in New Delhi India. The tweets here are analyzed thoroughly by doing sentiment analysis. However it uses a quiet different approach.

According to the writers the basic sentiments that a person might have are: Happiness, Anger, Sadness, Fear and Disgust. These are the sentiments that Paul Ekman and his team found in their research in 1972 to be the most common among human beings. This paper tried to extract the sentiments that users have when they tweet by using data cleaning, opinion mining and text analysis. This process was implemented in two basic steps:

1. Data preprocess. In this step the tweets were gathered and cleaned. By cleaning we mean that words irrelevant to sentiment analysis and opinion were discarded. The words that were considered valuable for analysis were adjectives, verbs and adverbs.

2. Value tweet. As we mentioned earlier there are 6 different sentiments that a word might represent. To value a tweet they tried to calculate with a linear method if the words within the tweet express sentiment and if so, how strong these sentiments are. There is a different scent of emotion in each word. Some words are considered "happier" or "angrier" etc. that others.

Methodology

The tweets were gathered using the Twitter's public databases and of course the Twitter's API. After collecting the tweets naturally the writers concluded that there were unnecessary information in them that made their job more difficult. Consequently, the data were cleaned by deleting useless symbols, punctuations, specific words and so on. In addition, hashtags, usernames, URLs and some Twitter symbols like "RT" were also removed. To retrieve the important words (adjectives, verbs and adverbs) a POS tagger was used (its name is NL linguistic parser). We can see below how exactly the data gathering and analysis was done.

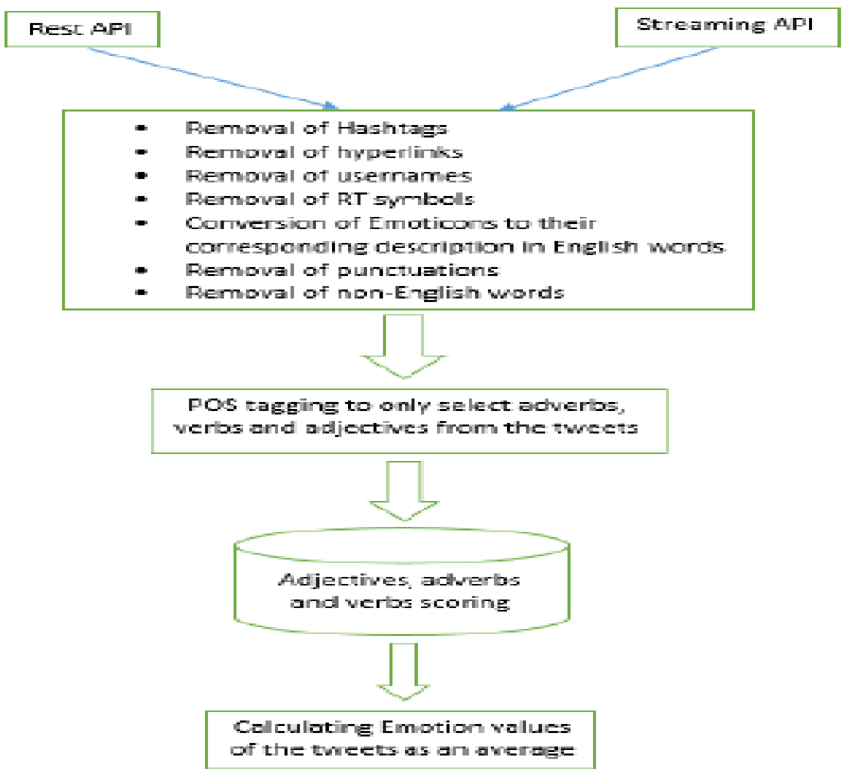
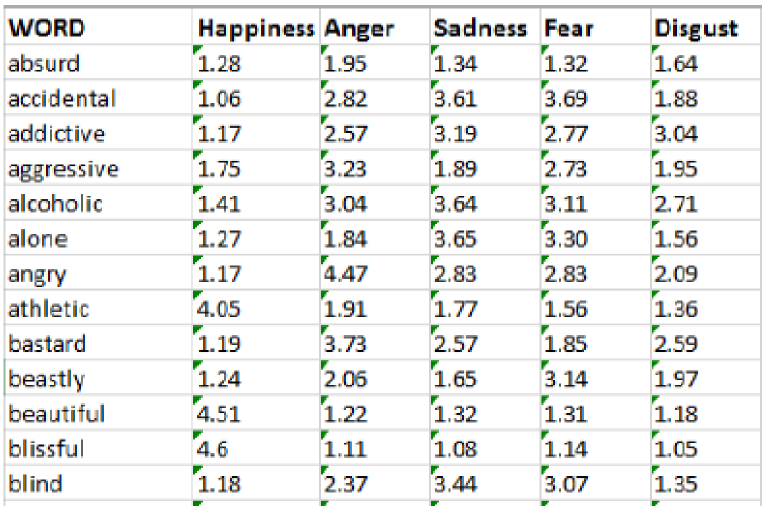

Rest API
Streaming API
Removal of Hashtags
Removal of hyperlinks
Removal of usernames
Removal of RT symbols
Conversion of Emoticons to their corresponding description in English words
Removal of punctuations
Removal of non-English words
POS tagging to only select adverbs, verbs and adjectives from the tweets
Adjectives, adverbs and verbs scoring
Calculating Emotion values of the tweets as an average

Figure 10: *data gathering and analysis stage [6]*

Each word has a sentiment behind it. However, most words may express more than one sentiment. This is why every word, in this case, has a scoring value. Using a novel corpus based method, each word has a vector for each sentiment it represents. Some examples of adjectives and their sentiment values are shown below.



WORD	Happiness	Anger	Sadness	Fear	Disgust
absurd	1.28	1.95	1.34	1.32	1.64
accidental	1.06	2.82	3.61	3.69	1.88
addictive	1.17	2.57	3.19	2.77	3.04
aggressive	1.75	3.23	1.89	2.73	1.95
alcoholic	1.41	3.04	3.64	3.11	2.71
alone	1.27	1.84	3.65	3.30	1.56
angry	1.17	4.47	2.83	2.83	2.09
athletic	4.05	1.91	1.77	1.56	1.36
bastard	1.19	3.73	2.57	1.85	2.59
beastly	1.24	2.06	1.65	3.14	1.97
beautiful	4.51	1.22	1.32	1.31	1.18
blissful	4.6	1.11	1.08	1.14	1.05
blind	1.18	2.37	3.44	3.07	1.35

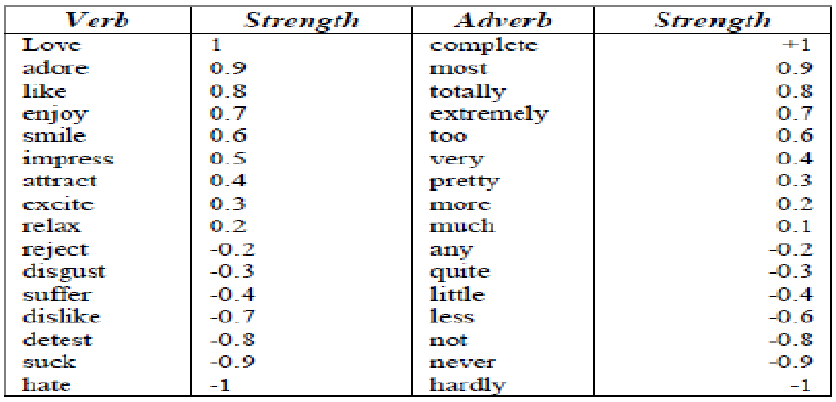
Figure 11: *sentiment values of words [6]*

Adverbs are also very important in opinion mining. An adverb behind an adjective can change the whole meaning of a sentence. A simple example to see how important adverbs are is the following: We have two sentences that their only difference is an adverb behind the adjective.

1. “The weather is nice today”

2. “The weather is not nice today”

The only difference in these sentences is the “not” before the adjective. Their meaning and sentiment are exactly opposite. This is a simple demonstration of how strong adverbs are in a sentence. Some examples of adverbs combined with verbs can be seen in the figure below.

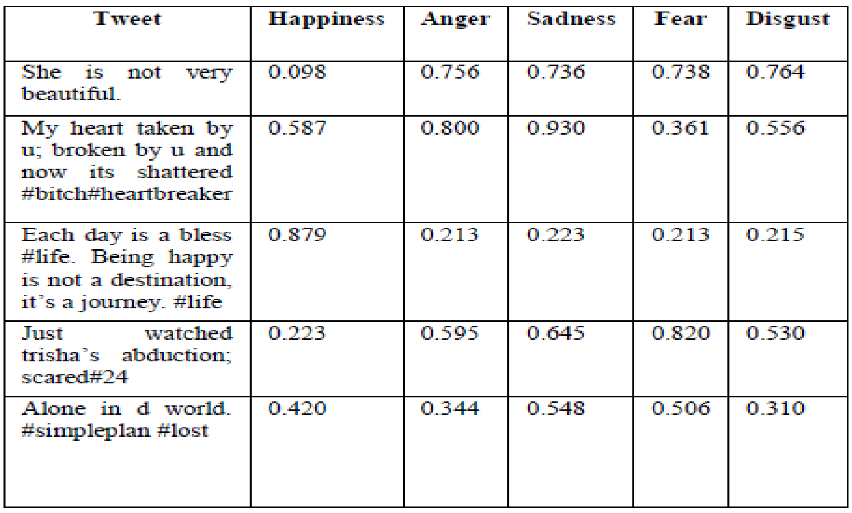


Verb	Strength	Adverb	Strength
Love	1	complete	+1
adore	0.9	most	0.9
like	0.8	totally	0.8
enjoy	0.7	extremely	0.7
smile	0.6	too	0.6
impress	0.5	very	0.4
attract	0.4	pretty	0.3
excite	0.3	more	0.2
relax	0.2	much	0.1
reject	-0.2	any	-0.2
disgust	-0.3	quite	-0.3
suffer	-0.4	little	-0.4
dislike	-0.7	less	-0.6
detest	-0.8	not	-0.8
suck	-0.9	never	-0.9
hate	-1	hardly	-1

Figure 12: *adverbs and verbs sentiment strength scores [6]*

After evaluating the words for each tweet we need to categorize it depending on the sentiment it expresses. The writers used a variety of scoring methods. However, the best results came from the following method: if the scoring value of a verb or and adverb was less than 0, it was considered negative and the value of the adjective that follows was subtracted from 5. If the value of the verb or adverb was greater than 0, it was multiplied by the value of the adjective that follows. Afterwards, the results that came up were added and their sum was divided by 5 times the number of the adjectives.

The whole procedure we described was then used in a use case. 1000 tweets were retrieved and cleaned in the way we saw earlier. Using POS tagging the adjectives, verbs and adverbs were collected. Afterwards, the words were valued individually, when referring to adjectives, or in groups, when there was a verb or an adverb before the adjective. Here are some samples of tweets and their values for each sentiment.



Tweet	Happiness	Anger	Sadness	Fear	Disgust
She is not very beautiful.	0.098	0.756	0.736	0.738	0.764
My heart taken by u; broken by u and now its shattered #bitch#heartbreaker	0.587	0.800	0.930	0.361	0.556
Each day is a bless #life. Being happy is not a destination, it's a journey. #life	0.879	0.213	0.223	0.213	0.215
Just watched trisha's abduction; scared#24	0.223	0.595	0.645	0.820	0.530
Alone in d world. #simpleplan #lost	0.420	0.344	0.548	0.506	0.310

Figure 13: *A sample of tweets and their sentiment score [6]*

Conclusions

The framework that this paper introduces is a simplistic but extremely efficient approach for sentiment analysis in social networks like Twitter. This application may find several different uses like recommendation systems in business intelligence and also to reveal vote intentions of social media users. However the use of sentiments like anger and happiness to extract knowledge is not the best idea. It makes the whole process more difficult than it is. In this dissertation tweets are going to be categorized as positive, negative or neutral.

2.6 Opinion Mining about a Product by Analyzing Public Tweets in Twitter [7]

In January 2014 T.K. Das, D. P. Acharjya and M. R. Patra from the VIT and Berhampur University published a paper about sentiment analysis for products using public tweets from customers. This paper took the topic one step further from the previous by building an application that helps researchers and business analysts to see how consumers feel about a released product.

Twitter is a social network where users express their opinion without any limitations and this makes it valuable for today's companies. The sentiment analysis done by the writers was based on text like the previous paper we saw, however here each tweet was characterized as neutral, positive or negative. This was done with the help of natural language processing and filtering. After a tweet was categorized, a report was generated for the high-end users in graph forms, pie charts or tables so that they could analyze the results and come up with valuable conclusions. The purpose of this procedure was to save both time and money for consumers and business analysts.

As we said earlier most of the researchers that want to retrieve tweets from Twitter use the REST API. However, here the writers did it with the Streaming API. To connect with the Streaming API they needed a stable and persistent connection. When a system is connected with the API it cannot respond to requests from users. That happens because the connection itself runs separately from the process that responds to HTTP requests. The Streaming process is the one that retrieves the tweets and filters them before storing them in the database. As the writers claim "The streaming API is better than the REST API because it is always better to have real time visualization of tweets".

Methodology

The approach introduced by the paper consists of five steps, the architecture of the system and a user interface. The five steps that describe this approach are:

1. Retrieve keywords from user. The user can type a keyword which might be a commercial product, a presidential candidate or any other topic. Based on the keyword the user provides the application retrieves all tweets related to that keyword.

2. Streaming tweets from the Web. After getting an access token by registering in Twitter, tweets are retrieved and filtered based on the keywords the user provided.
3. Sentiment Analysis. Here, tweets are analyzed with opinion mining. Every tweet has a score that defines it as positive, negative or neutral.
4. Store in Database. After being analyzed each tweet is stored in a database with their id, time sent, username and sentiment analysis score.
5. Report generation. After the 4 first steps are over a report is generated. This report is useful for high-end users and it’s usually a pie chart or a cumulative graph that helps high-end users to draw valuable conclusions.

The following figure shows how the whole system works.

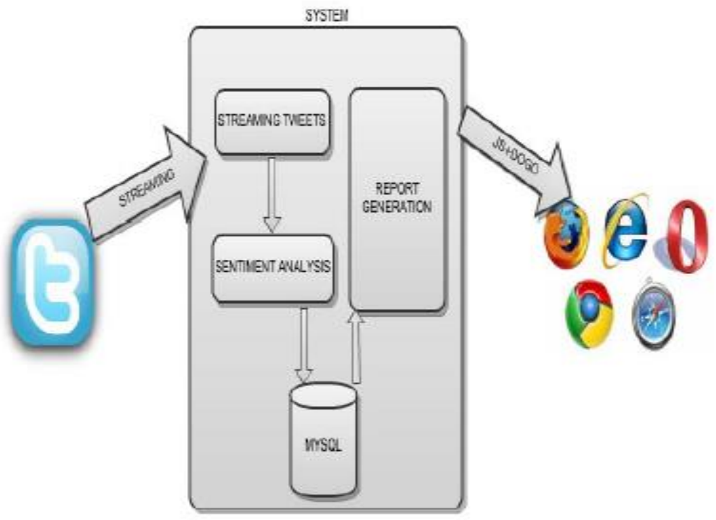
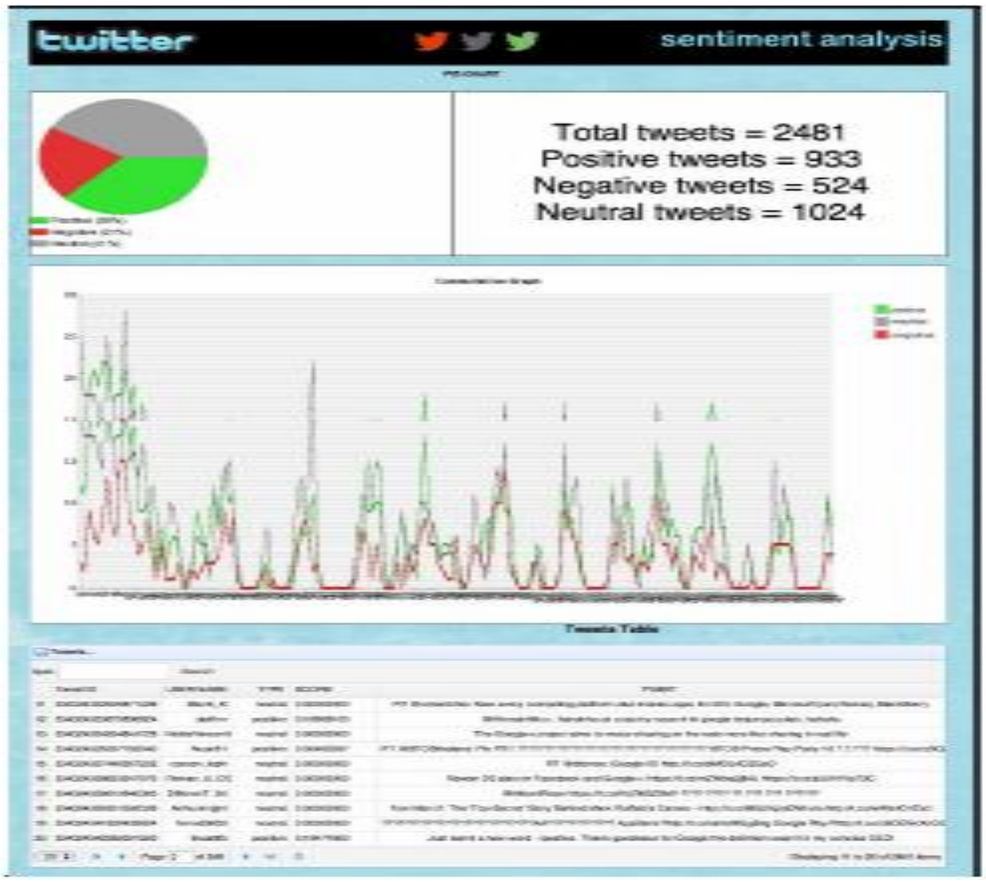


Figure 14: *How the system works [7]*

When the whole process is over a report is generated. As we mentioned earlier the reports came in three possible forms. The pie chart, which was a generic representation of the analysis, the cumulative graph that described how positive, negative and neutral feedback occured as a function of time and finally, the tabular representation that contained metadata of tweets like ID, type, date, username and so on. Below we can see an example of a report generated by the system.



Total tweets = 2481
Positive tweets = 933
Negative tweets = 524
Neutral tweets = 1024

Figure 15: *report generated by the system [7]*

Conclusions

The system proposed in this paper was a tool for conducting research about different topics (new products, opinion about election candidates etc.). What it did, was data gathering from the Twitter's streaming API and preprocessing in a way that sentiment analysis is easily done. The reports that were generated after the sentiment analysis of tweets was what mattered the most, because with their help high-end users (like managers, political analysts etc.) could draw conclusions and come up with useful ideas for future analysis. Despite the fact that the streaming API is more realistic, it might contain high risks when it comes to data storage. The REST API is more suitable for this dissertation and that is what we are going to use.

2.7 Literature review conclusion

We can see that a lot of researchers tried to do something similar to our project. We only mentioned the most popular, but there are more papers that tried to use Twitter to predict future events. This shows that this field has a bright future ahead since more and more analysts try to use social media as a tool for data mining. Every project we described had interesting perspectives and most of them were successfully implemented. Now we will see how our project was developed. We will continue with describing how we gathered our data and after that, how we analyzed it to perform sentiment analysis.

3. DATA GATHERING

In this chapter we are going to describe how the Twitter's API can be used to retrieve data and create a database of tweets. We had to make a decision on which API we will use. Twitter offers two options:

- The REST API
- The Streaming API

Our choice was to use the REST API. This decision was based on practical issues. If we chose the Streaming API we would have to deal with real-time data. We didn't the machine to support this. Also the Streaming API retrieves extremely large volumes of data in contrast to the Rest API which retrieves a smaller volume. In the following sections of this chapter we will analyze how both APIs work and see their advantages and disadvantages separately.

To communicate with the REST API we used PHP. In this chapter we will also demonstrate how we built the queries in order to get the data we wanted and also what parameters were used to do it. But first let's describe how the Twitter's APIs work.

3.1 REST API

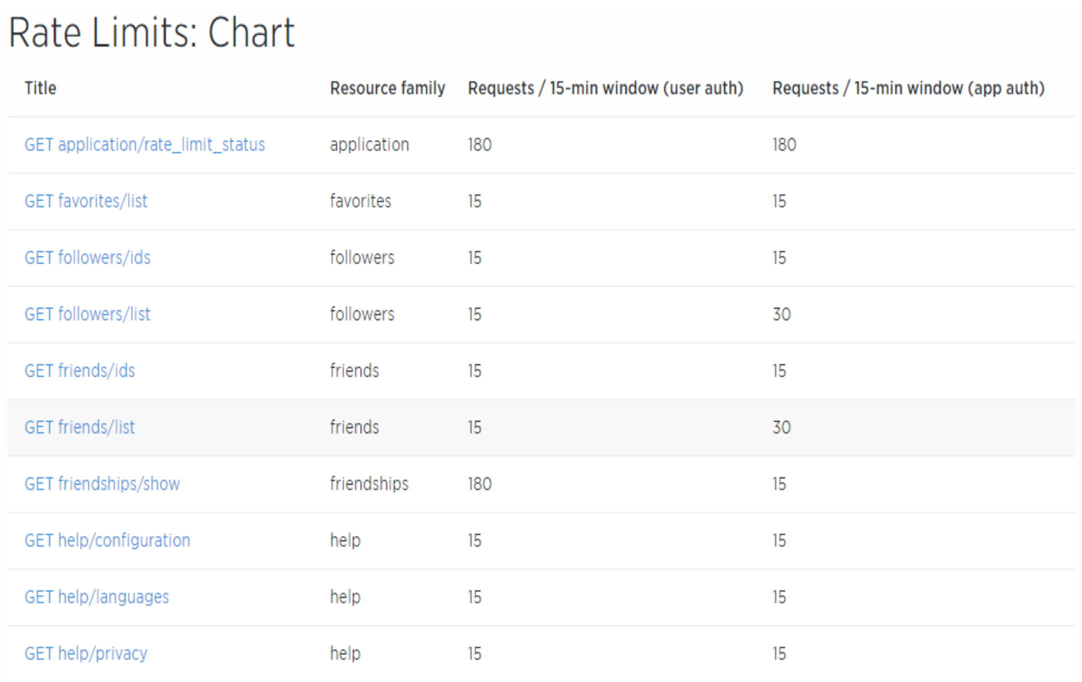
The Twitter's REST API is a powerful tool for developers that helps read and store data retrieved from Twitter. In order for someone to have access to Twitter's data, it is required to build an application that will provide us with authentication tokens in order to make our requests without any problems. The results that are retrieved from the requests are JSON files that are easily processed and analyzed. But as we said, before making the requests we have authenticate ourselves with the help of the application we built earlier. All requests have to be authenticated with *OAuth 1.a* or *Application-only Authentication [8]*.

OAuth 1.a authenticates users that want to download and process tweets. Its purpose is twofold: it authenticates the identity of the application that the user built and also identifies the permission that the user has in order to gain access to the APIs data. The second step is achieved by using an access token that is included in the URL of the query.

Application-only Authentication is an automated process for retrieving data. In this case the application makes requests by its own without the user manually controlling it. Of course, the API sets limits in the number of requests that can be made by an application so malicious users are not permitted to over-use the API. This way of authentication is based on the Client Credentials Gant flow of OAuth 2 specification. Before we continue we have to make clear that to make an *Application-only Authentication* an *OAuth 1.a* is also required.

3.1.1 REST API Rate Limits [9]

Rate limits for each user (or more specifically for each access token) are standard. If for example the allowed requests are 10 per rate limit window, it means that the user can make up to 10 requests per window for every access token he possesses. When the authentication model is *Application-only*, the limits for requests are determined for the entire application. Again, if the allowed requests are 10 for every rate limit window this means that 10 requests can be made for the rate window available from the application as an entity. The rate limits are separated with a 15 minute interval. This means every 15 minutes the rate limits are reset and the user can make new requests from scratch. We have to indicate once again, that throughout this process authentication is always required. Below we provide a table for requests that shows the rate limits for each one of them. The image is from the Twitter's developer forum.



Rate Limits: Chart

Title	Resource family	Requests / 15-min window (user auth)	Requests / 15-min window (app auth)
GET application/rate_limit_status	application	180	180
GET favorites/list	favorites	15	15
GET followers/ids	followers	15	15
GET followers/list	followers	15	30
GET friends/ids	friends	15	15
GET friends/list	friends	15	30
GET friendships/show	friendships	180	15
GET help/configuration	help	15	15
GET help/languages	help	15	15
GET help/privacy	help	15	15

Figure 16: *table of rate limits for every request [9]*

3.1.2 Search API

The Search API is a sub property of the REST API. What it does is to allow users-developers to search for specific tweets with the help of queries. The timeline of the data gathered is 7 days before the query is executed. For example if we make a query on 17/10/2016 at 5:00 am, the data that will be fetched are going to be from 10/10/2016 until the time we executed the query. It is very important to make clear that not all tweets of that timeline are retrieved. Just a sample of tweets that the API selects. The Search API is based on how relevant a tweet is to the query not on complete gathering of tweets relevant to the query. That means that some users and tweets will be missing in the results.

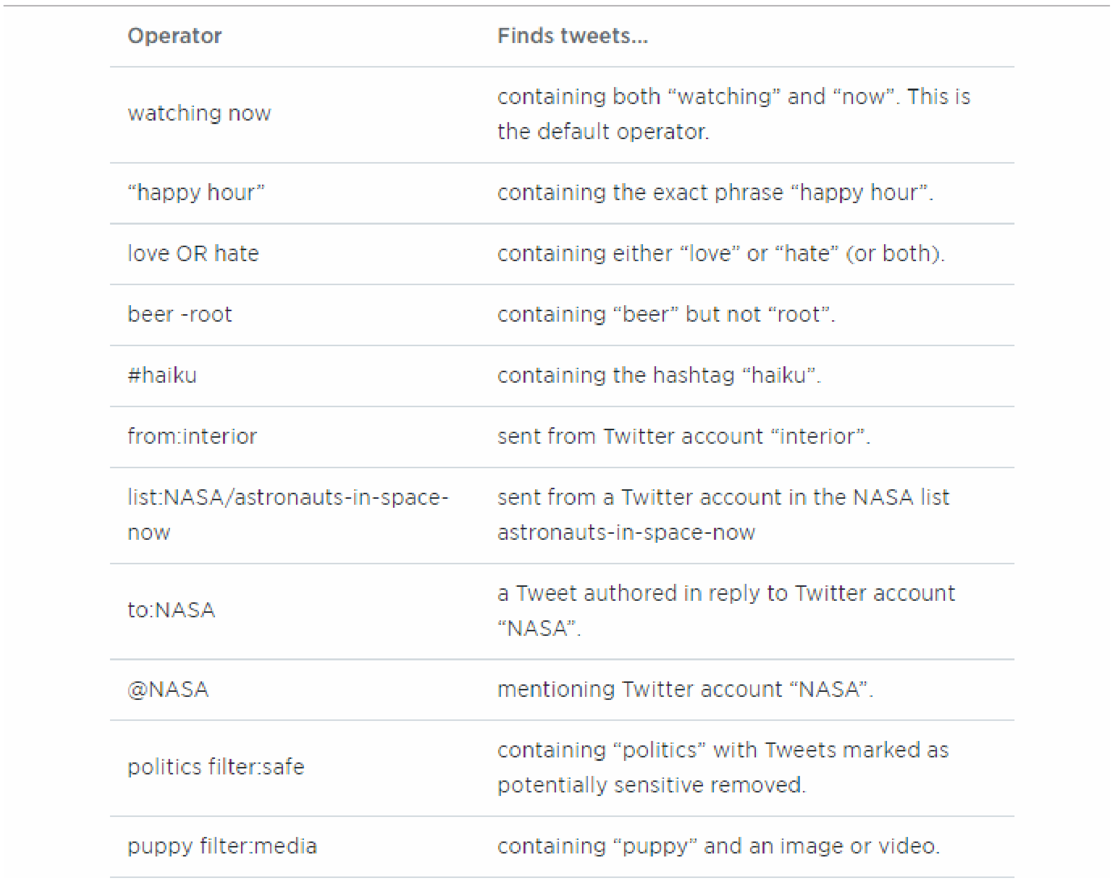
Queries are executed in URLs. To make sure our query is well-designed we can use the *search* option [10] of Twitter itself or the *Twitter advanced query search builder* [11].

Let’s see an example to understand how queries work. We want to retrieve tweets that refer to Barack Obama’s Twitter account (@BarackObama). Firstly, we run the search on twitter.com/search. When the results come up we get the URL that refers to our search. In our example we get the following URL:

https://twitter.com/search?q=Barack%20Obama&src=tyah

We copy the URL and now we have a query. Afterwards, we replace the twitter.com/search with https://api.twitter.com/1.1/search/tweets.json, so now we have https://api.twitter.com/1.1/search/tweets.json?q=Barack%20Obama&src=tyah. After logging in the application we have built earlier, we hit the URL and wait to get the results. At this point we have to remember that the twitter.com/search retrieves data without having a specific timeline while the Search API retrieves data from the past week, so the results will have some differences.

In order to execute queries efficiently we need to master operators and parameters. The operators are a useful tool for developers that helps customize queries depending on what information have to be gathered. Below we can see a list of example operators taken from Twitter’s developer webpage [12]:



Operator	Finds tweets...
watching now	containing both “watching” and “now”. This is the default operator.
“happy hour”	containing the exact phrase “happy hour”.
love OR hate	containing either “love” or “hate” (or both).
beer -root	containing “beer” but not “root”.
#haiku	containing the hashtag “haiku”.
from:interior	sent from Twitter account “interior”.
list:NASA/astronauts-in-space-now	sent from a Twitter account in the NASA list astronauts-in-space-now
to:NASA	a Tweet authored in reply to Twitter account “NASA”.
@NASA	mentioning Twitter account “NASA”.
politics filter:safe	containing “politics” with Tweets marked as potentially sensitive removed.
puppy filter:media	containing “puppy” and an image or video.

Figure 17: *table of operators [12]*

The parameters allow developers to control the results they get and set restrictions. There are four main parameters that can be inserted in a query. Those are:

1. *Geolocalization.* This is a parameter that allows us to search for tweets from a particular area. What is really impressive about this parameter is that we can actually define an area with coordinates that will be in the URL. When we define the coordinates the Search API will try to find tweets from users that their profiles are connected with the area. This is very helpful in our case as we will see later on.
2. *Language.* We can also restrict results to be in a specific language we define in the query.
3. *Result type.* With this parameter we can define weather our results will be the most popular or the most recent tweets. We can set the results to be a mix of both.
4. *Iterating in a result set.* There are some parameters that allow us to iterate within the results we have. Parameters like *until, since_id* and *max_id* help us navigate through the results. When we make a request from the API it is most likely to get a huge amount of data so iterating and analyzing results is a very challenging task as we will see later on.

Now let's see an example of a query that uses some of the parameters we described. We want to fetch all recent tweets from the Maracanã stadium in Brazil about the football match between Brazil and Germany. The query (always in URL) will be: https://api.twitter.com/1.1/search/tweets.json?q=&geocode=-22.912214,-43.230182,1km&lang=pt&result_type=recent

After testing both the REST API and its sub property, the Search API, we decided to make our requests for this project with the Search API. The options, parameters and operators that it offers are closer to our needs for this project so our decision was easy. However, when we get the results after a query we might have problems with the paging process. Let's see what we mean when we refer to the term "paging problem".

3.2 Paging problem

As we mentioned earlier, there is a limit on how many data a user can retrieve with one request and that is because there is an unlimited amount of tweets being generated every second. It is impossible for a machine to retrieve all tweets related to a query in few minutes. Limits help utilize the data gathering process by defining timelines. However, the volumes of data are still chaotic and that makes the storage of the data an extremely challenging process.

In this section we are going to describe what happens while and after the query is finished. How do we have to organize the results in order to avoid compromising the real-time nature of the social media? Which is the optimal way to place the results in pages?

To demonstrate how the storage and paging problem should be dealt with, we will see an example. Let's say that we have two pages of tweets. Each page has 5 tweets chronologically sorted (most recent come always first) containing also the each tweet's data like username, hashtags, retweets count, time posted, day posted, likes count etc. When we make the first request the first page is fetched. To get also the second page we have to make a second request. Now, let's assume that we want to add two more tweets in our dataset that are chronologically placed between the first and the second page. Now the first and the second pages will contain some same tweets. To understand the problem more easily, let's say that in the previous example we have 5 (or more) new tweets that are supposed to be placed between the two pages. Now all requests that will occur afterwards will fetch all tweets from the first request. This is a simple example of redundant data. There are two parameters that can be used in order to resolve this problem:

- The max_id parameter and
- The since_id parameter

3.2.1The max_id parameter [13]

The most common solution for the paging issue we described earlier is working with streams of data. This technique is called cur soring. Instead of reading the info of the first tweet in the list we should read data (mostly the IDs) from all the tweets we have already processed. Here is where the max_id parameter becomes a part of the game.

In order to use this parameter properly the first request to the API should be about the count of the tweets that each page will contain. The calls that will occur later on are the ones that will retrieve all the data we want. We need to keep the tweet with the lowest ID within the dataset. The tweet with the lowest ID is the most recent one. We set the value of the max_id as the value of the lowest ID for the next requests. When we fetch a tweet that has a lower ID than the value of the max_id, the max_id value changes and gets this ID.

The max_id parameter is considered inclusive. That means that it will always return the value of the lowest ID no matter how many times it did it again. As we will see later this is the method we chose to deal with the paging issue. Now let's see another option that can be used for the same cause.

3.2.2 The since_id parameter [13]

A more efficient way of dealing with the paging problem of the dataset is the since_id parameter. When working with a timeline, applications need a few moments to gather and process the new tweets that come up every time a call is made to the API. To make this process easier we can use the since_id parameter.

Let's see an example. We have 10 tweets that have already been processed by the application. Now let's say that after processing the first 10 tweets, we have new tweets that need to be processed. Those tweets are 11-18. The most common error here is to process all tweets from the beginning and probably end up with duplicates in the results.

To resolve this problem we introduce the since_id parameter. The initial value of the parameter is set as the highest number of the IDs (of tweets) that we have already processed. The difference with max_id is that since_id is not and inclusive parameter, which means that we don't have to make any changes to the results.

After applying the since_id parameter to our dataset, our application will be able to compare the value of the parameter to the ID of the new tweets and return only those with higher values.

At this point we have to make clear that the optimal solution to the paging problem is a combination of both parameters. Unnecessary data is a serious problem and these two parameters can be used to avoid data redundancy.

3.3 Streaming API

An alternative that Twitter developers have to retrieve data is the Streaming API. As we saw in the Literature Review chapter a lot of researchers chose this API to gather their data. What is different here is that it offers a low latency access to the data of the API. The Streaming API offers a real-time experience and since Twitter is a social media with constantly new information and data, this is an option to be taken into serious account.

There is a variety of stream points that Twitter offers. Every single one of them is designed to satisfy different needs. The basic streams are the following:

1. *User streams*. Retrieving all data from a particular user and also data that are connected somehow to the user
2. *Public streams*. Streams of public data that help developers fetch information about different users, tweets, topics and hashtags
3. *Site streams*. It is about servers that need to be connected on Twitter on behalf of different users.

Differences between the REST and Streaming API

It is easy to see that there are a lot of differences between the REST and the Streaming API. Firstly, we have to point out that the results we get after making a request with the REST API are quite different from the results of the Streaming API. That is because the Streaming API retrieves all tweets related to the query we gave. This means that queries will take a lot of time to be executed and the volumes of data will be extremely big. In addition, the Streaming API needs a continuous HTTP connection which makes it even more difficult to make requests without crashes. On the other hand, The REST API retrieves only a sample of tweets that match the query. Next, we are going to see what decisions we made for this project: which API we chose and how we gathered the tweets we needed for this project.

3.4 Our approach

After demonstrating how someone can use the Twitter's APIs to retrieve tweets we are going to describe the decisions we made about the data gathering process.

The first decision we had to make was to choose which API we will we use to make our requests. The Streaming API gave a real-time tone to the project but the massive volume of data that we would have to process was an issue for our hardware equipment. The REST API on the other hand, was a choice that offered a sample of tweets for every query we would make. Thus, the volume of data was more feasible to handle. As a result, we decided to go with the REST API and more specifically its sub property, the Search API. In order to contact the API we wrote our code in PHP. As we will see later PHP is very effective when dealing with APIs.

After choosing our API we had to build our own application that will authenticate our requests and make it possible to make queries and calls with the permission of Twitter. We named our application TweetGrabber13. The process to build this applications was extremely simple as there are guidelines from Twitter explaining step by step exactly how to do it. Now let's see what we did code-wise in order to get the data we needed.

3.5 PHP code review

When dealing with APIs the best programming language to use is PHP. Compared to other programming languages PHP is easier to understand for beginners because it is built for lower level programming. Also, the documentation PHP provides is very detailed and thus, very helpful when we will have to deal with possible code anomalies. In addition, PHP offers a very powerful extension called Client url (curl). This extension was created by Daniel Stenberg and it helps our systems establish connections with various types of servers that operate with different protocols. For more information about this extension you can visit http://php.net/manual/en/book.curl.php [14]. In our case, in order to connect to the Twitters' API and make the request we want to retrieve tweets, we used the curl extension.

The most important task to do before establishing a connection with the Search API is to authenticate ourselves as developers, using the credentials we got from the application we built (TweetGrabber13). The authentication process is extremely complicated code-wise, so we decided to use a free available repository from Github.com. This repository belongs to J7mbo (https://github.com/J7mbo [15]) and it is named twitter_api.php. The authentication process is done once we insert our authentication tokens to the code. After that, we are ready to start making requests to the Search API.

As we discussed earlier, in order to get the results we want, we have to add some parameters to the code, making the query more targeted. The parameters we placed in our code are the following:

- Query string (hashtag). Here is where we declare the hashtag that we want to get tweets for. What we type in this field will be the string characters that every tweet we fetch will contain. For example '#DonaldTrump'.
- Query string (date and time). Here is where we declare the exact date we want the tweets to be from. We have to define a time lapse of days. For example 'since: 2016-10-21 until: 2016-10-22'.
- Language. With this parameter we declare the language we want the tweets to be in. In our case we chose to retrieve tweets that are in English.
- Coordinates. Here we had to declare a specific geographical location we want the tweets to be from. As we said we targeted the three most important states for the elections. In order to find the coordinates we want we used https://www.daftlogic.com/projects-google-maps-area-calculator-tool.htm [16]. In this site gives us the ability to define an area and get the information we want about it (perimeter, longitude). Let's see an example:

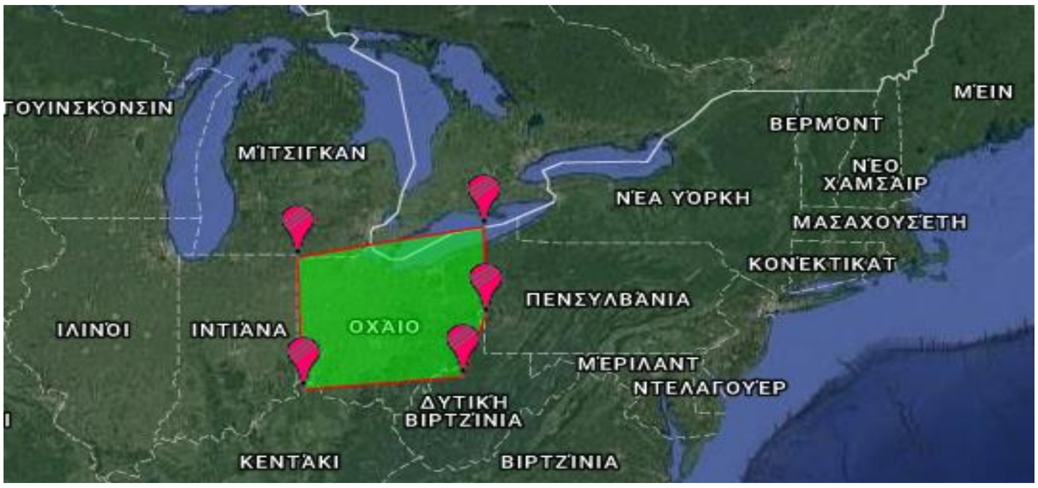
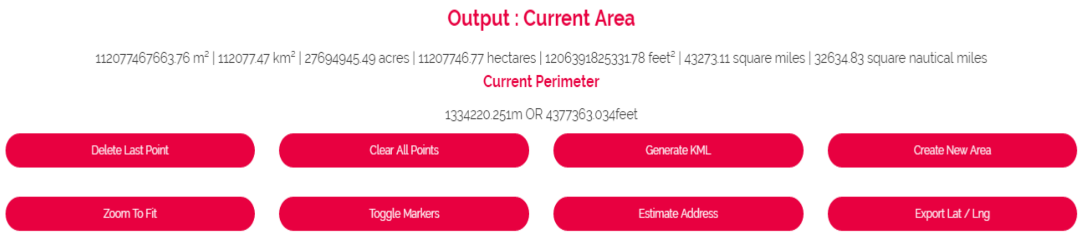


Figure 18: *google maps area calculator (https://www.daftlogic.com/projects-google-maps-area-calculator-tool.htm [16])*

In the image above we set an area we want tweets to come from. This tool is really easy to use. We just have to define the location we want by setting points on the map, and the application provides us with all the information we need. The image below shows all the information that the application provides.


Output : Current Area

112077467663.76 m² | 112077.47 km² | 27694945.49 acres | 11207746.77 hectares | 1206391825331.78 feet² | 43273.11 square miles | 32634.83 square nautical miles

Current Perimeter

1334220.251m OR 4377363.034feet

Figure 19: *Google maps area calculator, information about an area (https://www.daftlogic.com/projects-google-maps-area-calculator-tool.htm [16])*

In order for the code to work properly, we need to insert the perimeter of the area we defined and afterwards get a geographical point that is close to the middle of the area we defined with the help of https://www.google.gr/maps [17]. Below we demonstrate an example.

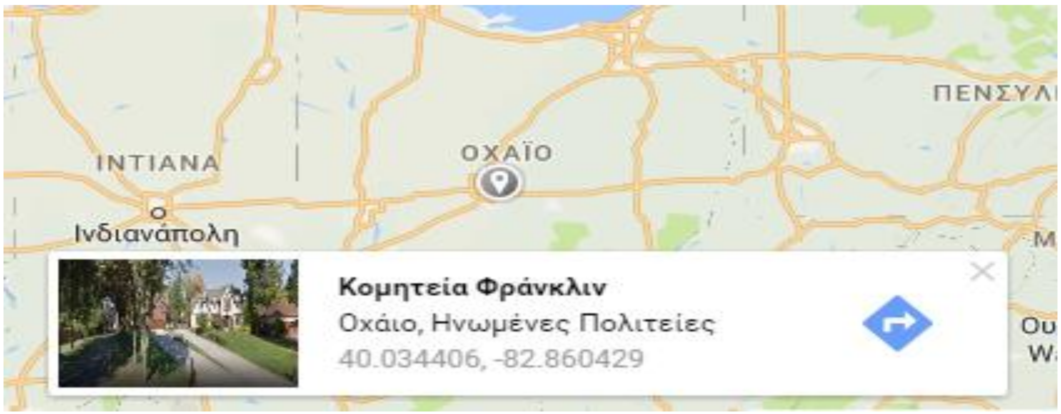
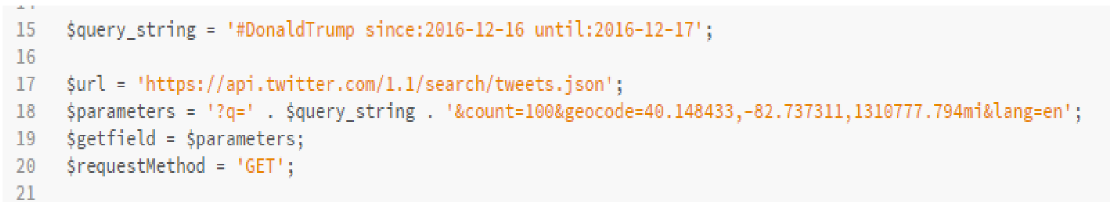


Figure 20: *coordinates of a point in Ohio taken from* https://www.google.gr/maps *[17]*

Besides the perimeter of the area we also need the coordinates of a random point that has to be as close to the middle of the area as possible. So the coordinate parameter needs exactly three numbers:

1. Perimeter
2. Latitude
3. Longitude

Below we provide an sample image of the code we wrote in PHP, demonstrating how we used the parameters we described in a query:



```php
$query_string = '#DonaldTrump since:2016-12-16 until:2016-12-17';

$url = 'https://api.twitter.com/1.1/search/tweets.json';
$parameters = '?q=' . $query_string . '&count=100&geocode=40.148433,-82.737311,1310777.794mi&lang=en';
$getfield = $parameters;
$requestMethod = 'GET';

```

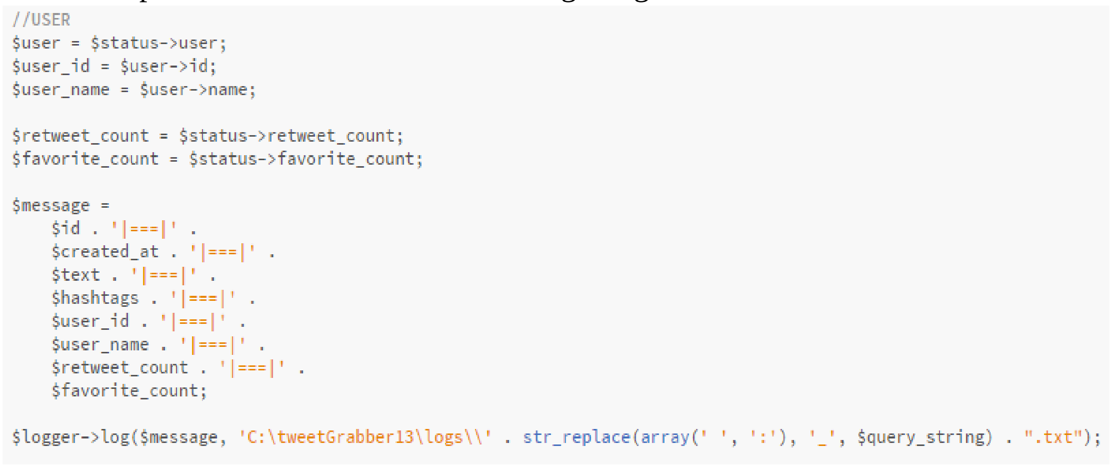
Figure 21: *Query we designed to get tweets from 16/12 until 17/12 with the hashtag #DonaldTrump from the state of Ohio*

As we also mentioned earlier, a very common issue when making requests to the API is the paging problem. We decided to use the max_id parameter to avoid getting errors. For every request we make, each page will have a maximum number of 100 tweets. Each tweet has an ID that uniquely identifies it. The IDs are placed in a descending order. The last tweet of each page is the tweet with the lowest ID number. Let's assume that we make a request and get tweets with IDs between 1000 – 801. With the code we built, the first page will have the tweets 1000 – 901 and the second page will have to check which the last ID of the previous page and get tweets with IDs lower to this. So in our example the second page will have tweets with IDs 900 – 801.

The tweets are downloaded in JSON format. With our code we convert them to txt. The tweets contain tones of information. We have to clear tweets from unnecessary information in order to make them simpler to analyze. We decided to keep only the information that will help us in the sentiment analysis process. Those information are:

- ID. We need the ID for the paging process
- Created_at. This is the date and time of the tweet. We need this information to see when this tweet was posted.
- Text. This is the actual content of the tweet.
- Hashtags. We set a separate field for the hashtags the tweet includes in order to see if it is relevant to what we want.
- User_id. The id of the user that posted the tweet
- User_name. The username.
- Retweet_count. This field shows us how this tweet was received by other users. The more retweets the twee has the more popular the opinion it expresses is.
- Favourite_count. This field shows us how many users liked the tweet. It is important for the same reason as the Retweet_count

Below we provide a screenshot of the code getting the data we mentioned:

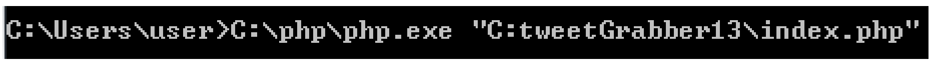


```
//USER
$user = $status->user;
$user_id = $user->id;
$user_name = $user->name;

$retweet_count = $status->retweet_count;
$favorite_count = $status->favorite_count;

$message =
    $id . '|===|' .
    $created_at . '|===|' .
    $text . '|===|' .
    $hashtags . '|===|' .
    $user_id . '|===|' .
    $user_name . '|===|' .
    $retweet_count . '|===|' .
    $favorite_count;

$logger->log($message, 'C:\tweetGrabber13\logs\\' . str_replace(array(' ', ':'), '_', $query_string) . ".txt");
```

Figure 22: *getting rid of unnecessary information the tweets have, keeping only the important information*

A very important factor we need to take into account is the API limit. We mentioned that the API doesn't let more than 180 requests for every 15 minutes. To deal with this we set a timeout of two seconds between requests in order to avoid getting error messages while collecting data.

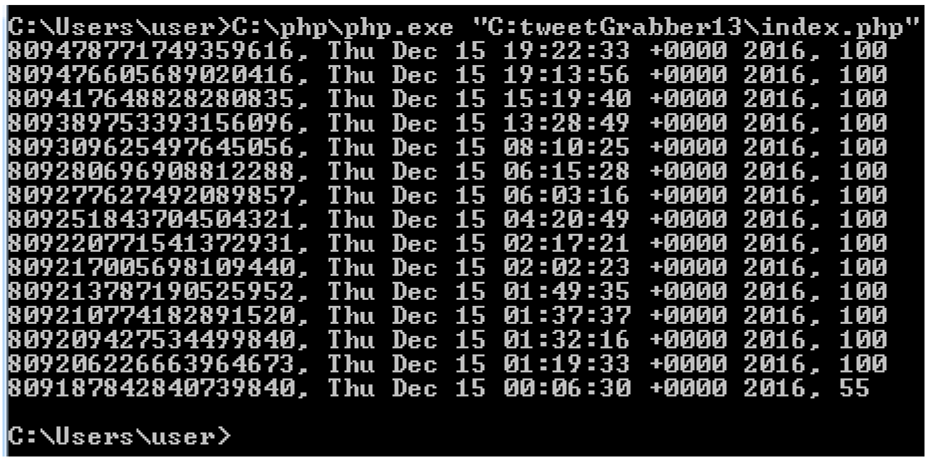
The PHP should be executed in command line as shown below (the paths are different depending on the system).



```
C:\Users\user>C:\php\php.exe "C:tweetGrabber13\index.php"
```

Figure 23: *how to execute the PHP code in command line*

We built the code in a way that while running it provides information about the data gathered so far. The following images shows an example of a request we made while the script was running.



```
C:\Users\user>C:\php\php.exe "C:tweetGrabber13\index.php"
809478771749359616, Thu Dec 15 19:22:33 +0000 2016, 100
809476605689020416, Thu Dec 15 19:13:56 +0000 2016, 100
809417648828280835, Thu Dec 15 15:19:40 +0000 2016, 100
809389753393156096, Thu Dec 15 13:28:49 +0000 2016, 100
809309625497645056, Thu Dec 15 08:10:25 +0000 2016, 100
809280696908812288, Thu Dec 15 06:15:28 +0000 2016, 100
809277627492089857, Thu Dec 15 06:03:16 +0000 2016, 100
809251843704504321, Thu Dec 15 04:20:49 +0000 2016, 100
809220771541372931, Thu Dec 15 02:17:21 +0000 2016, 100
809217005698109440, Thu Dec 15 02:02:23 +0000 2016, 100
809213787190525952, Thu Dec 15 01:49:35 +0000 2016, 100
809210774182891520, Thu Dec 15 01:37:37 +0000 2016, 100
809209427534499840, Thu Dec 15 01:32:16 +0000 2016, 100
809206226663964673, Thu Dec 15 01:19:33 +0000 2016, 100
809187842840739840, Thu Dec 15 00:06:30 +0000 2016, 55

C:\Users\user>
```

Figure 24: *the information we get while executing the PHP script*

This script was executed multiple times in order to gather all the data we needed. We gathered 277.509 tweets in total and the script run without getting any errors. The data gathering process lasted for about a month. We made daily requests collecting as many data as possible. We encourage anyone to run the script in their own systems to see how easy and sophisticated the gathering process is, with the code we wrote. Next we are going to describe how we analyzed the tweets we gathered in order to make our predictions for the elections.

4. Sentiment analysis

In this Chapter we are going to describe how we analyzed the tweets we gathered, in order to classify them as positive, negative or neutral. The term that defines what we did exactly is sentiment analysis (or as some call it opinion mining). Before describing our implementation process we will first get familiar with some sentiment analysis terms and see how we used every feature we had to make the implementation phase as sophisticated as possible.

4.1 What is Sentiment analysis?

Nowadays, Internet and more specifically social media, are considered to be one of the most powerful tools available for data analysts. Via the internet, users can make purchases, read reviews about products, movies, events etc. After finishing their activity, most of them have the urge to express their opinion by sharing their experiences with their online friends. There are many ways for someone to express their opinion while being online. Most websites offer comment sections, forums etc. However most of the users choose to express their sentiments by posting something in their favorite social media. This information is very important for corporations which would like to know what the public opinion about a specific subject is. New products, movie critics and in our case election candidates would definitely want to gather feedback and analyze how the crowd reacts to their every move.

Scott Cook, a well-known entrepreneur once said: “A brand is no longer what we tell a customer it is – it is what customers tell each other it is”. By that we understand that a positive public sentiment expressed in social media can have an extremely positive effect for the brand (or whatever the corporation is about). In our time, businesses choose to hire people that their only job is to monitor social media and come up with conclusions about what people think about their products. Of course, the amount of data is huge so analysts try to perform automated analysis by parsing the reviews and extracting information about the feelings of the consumers or in our case voters. This is the idea behind the term Sentiment Analysis.

To be able to perform sentiment analysis we need to understand the term of Natural Language Processing (NLP) which is the main feature of this field. The purpose is to extract subjective information from pieces of text. In the case of the USA elections that will occur on November 8th, we are going to perform sentiment analysis about the two dominant candidates Donald J. Trump and Hillary Clinton on Twitter. We have collected data from Twitter and we going to see how people respond to debates, speeches of the two candidates and so on.

This problem is definitely a classification problem. We want to classify tweets that refer to the candidates as positive or negative. To be more specific we are going to determine the polarity of tweets or as some call it, the semantic orientation of the text data we gathered. There are different approaches in which this problem can be addressed. Actually there are tons of different approaches. One of the main challenges of sentiment analysis is to choose the correct approach, depending on what the nature of the problem we are dealing with is. There might be a case where the design and implementation process of a model is correct, but the solution proposed in not the optimal for the specific problem. In sentiment analysis we have two main approaches that analysts use to address this problem:

- *Rule Based Approach.* This approach requires to study vocabulary and a lot of lexical features in order to come up with rules that will later be applied in the dataset we acquired. Human judgement is very important here since the rules will be defined by the analyst that performs the data preprocess. The rules that we will come up with, will determine the polarity of the textual data we have.

- *Machine Learning approach*. It is based on machine learning techniques and statistics. In this approach the human factor is less important since we don't have to give the rules beforehand. However, a training dataset with data that are already classified as positive or negative is required. With the help of our training data we train a classifier that will later help us classify our test data.

4.2 Machine Learning Approach

We already mentioned that the purpose of sentiment analysis is to determine the polarity or the semantic orientation of a text document. The best way to do that is a Machine Learning approach. There is also the Rule based approach but we decided that the best way to complete this project is Machine Learning. There are methods that are based on Machine Learning that can help us address this problem. The key term here is classification. The target is to classify every tweet as positive, negative or neutral. There are two basic methods that can be used for classification:

- Naïve Bayes classification method
- Support Vector machines method

There are other methods as well but those two are considered as the most efficient and simple ones for this particular problem. It is not clear which method is the most suitable in our case. This is a subjective issue and every researcher has to make a series decisions that have to do with methodology and approach.

Now there are some questions that have to be answered before starting a project based on a Machine Learning Approach (ML). As we mentioned earlier, in order to solve a problem with a ML approach we have to obtain a set of training data for our classifier. This is very important because all the results that will come up at the end of the project will be based on this training set we have. So, one of the most challenging tasks in Machine Learning is to choose the correct training dataset. But what is training data actually? It is a dataset with predefined classes that all its data are already assigned to a class. So in our case we will need a dataset of tweets that are already classified as negative, positive or neutral. After getting this data we will analyze the data and see potential patterns and rules that come up and use it in order to classify the tweets we get afterwards. It’s very easy to find datasets that are already classified. There are several human annotated corpora available from well-known researchers and universities that are available for anyone to obtain. For example, a researcher called Niek Sanders has manually labeled 5000 tweets as positive or negative and put the classified dataset online for everyone to download. In addition, the University of Michigan published a similar file with classified texts that was for a competition on Kaggle about Sentiment analysis. As we will see later, in our implementation process we got a combination of the two corpuses we mentioned that can be found on http://thinknook.com/twitter-sentiment-analysis-training-corpus-dataset-2012-09-22/ [18].

Another important and challenging task is the algorithm we will choose to go with. This process needs to dedicate a lot of time in the testing process and observe which algorithm gives the best results. Depending on what algorithm we chose to go with, our classifier will “learn” how to classify. Of course, that also depends on what features we want it to contain. We have to choose if we want to divide the text in phrases or in single words and determine how the classifier will draw conclusions from the dataset. The most common way to do this is to look at the document and each word in it and afterwards represent it as a combination of the most “important” words it contains. By important we mean words that express sentiment like “love”, “hate”, “admire”, “dislike” etc.

After some tests we run we decided that the model we will go with is Naïve Bayes Model. We found out that SVM is better for large texts and since we will deal with tweets that are limited in terms of the number of words they contain, we decided to use Naïve Bayes model.

4.3 Data preprocess

Before we start the sentiment analysis part, we need to preprocess our data in order to make them simpler to process. Sentiment analysis is based on patterns that make natural language expressions easier to analyze. Let’s see some examples of tweets that we will have to deal with while collecting our data:

“#DonaldTrump his defence programme is right ON POINT! He is the leader to go with!”

“Today Hillary Clinton made is scheduled to speak to the feminist club of Wisconsin #ImWithHer”

There are some patterns that make these tweets special. It is clear that in the first tweet we have a strong opinion that is in favor of Donald Trump. We can say that because of some features that are a part of the tweet. For example the capital letters followed by the exclamation point “ON POINT!” is an attempt by the user to emphasize and make his statement more powerful and clear. The topic that every tweet speaks about is the hashtag (#). Any conclusions we come up with when processing a tweet is about the hashtags it contains. For example the first tweet we saw has the hashtag #DonaldTrump, so the particular tweet is expressing an opinion about the presidential candidate Donald J. Trump. Consequently, what we will do is collect data that have specific hashtags that are about the two presidential candidates and extract sentiments that users try to express when typing a tweet.

One of the most important challenges in sentiment analysis is to find a way to deal with misleading information. There might be cases where a user misspelled a word or instead of expressing direct opinion he chose to be sarcastic about the topic. In order to avoid jeopardizing the results we have to search for patterns that will fix anomalies within our dataset and make it easy and simple to process. Here is where regular expressions come in. Regular expressions help us look for patterns that we might want to correct, substitute and so on. More specific, regular expressions are sequences of characters that define a search pattern of natural language expressions. A very crucial term that we have to get familiar with is operators. Operators, are symbols that help us build a search pattern and find anomalies in texts. When our pattern is expressed with regular expressions, we can use it to find words that this pattern is applied on.

Let's take a look at how operators work and which operators will be helpful in our case. When we look for a specific word, operators are not necessary. So, to search for the word "love" we don't need anything complex, we just perform a simple query and the word we want is retrieved. However, in some cases we might have to look for something a little vaguer. Let's say that we want to look for one word or another one (not both just one out of the two words). For example we want to look for a text that contains the word "love" or the word "hate". Here we can use the "|" character. This character can be used to express alternatives. So the regular expression we want here is "love|hate". This is very helpful because most programming languages, like python (which is the language we will be using), do not have built in operators for these kinds of situations, so we would have to write many lines of code in order to describe this regular expression. The "|" operator can be used in many ways and it's important to learn to use these operators in many ways in order to achieve optimal results with minimum effort. For example if we want to find a pattern that contains the word "build" or "built" we can use the "operator" like this "Buil(t|d)". That way we make a pattern that returns the word "Built" of "Build".

We will demonstrate a few more examples of regular expressions in order to get familiar with patterns and word analysis. Let's assume that we want to see if a pattern repeats itself and if so, how many times it is repeated. To help us with this process we will have to use quantifiers. The quantifiers we will be using are the following: ?, +, *, {m}, {min, max}. Each one of these quantifiers serve a purpose and help us perform tasks that are useful when we want to find patterns that repeat themselves and try to fix any text anomalies. The "?" quantifier is used when we want to represent 0 to 1 instances of the previous character. For instance, we have the regular expression "fou?r". The quantifier used indicates that we might get results where the "u" letter occurs once or not at all. So the words we are going to get are both "four" and "for". The "*" quantifier is used when we want to represent 0 or more instances of the previous character. The regular expression "fou*r" includes the word "for", "four", "fouur", "fouuuuuur" and so on. The "+" quantifier is used when we want to represent at least 1 or many instances of the previous character. So, the regular expression “fou+r” includes the same as “fou*r” accept the word “for”. When we want to indicate that a character can appear exactly *m* times we use the {m} quantifier. An instance of this case is “fou{3}r”. Here we are going to get the word “fouuur” only. If we want to indicate that a character will appear at least *m* times we use the following syntax “fou{3,}r”. Here we will get words with 3 or more “u” (fouuur, fouuuur etc.). When we want to find a pattern that occurs a specific number of minimum instances and a specific number of maximum instances we use the {min, max} quantifier. For example “fou{1, 3}r” will return the following words: “four”, “fouur”, “fouuur”.

The operators and quantifiers we described will help us implement our sentiment analysis model. Of course, there are a lot of ways to use them and combine them in order to get the results we need. In our model we used complex regular expressions that help optimize the process of opinion mining and deal with various text anomalies that occur in tweets.

Next we are going to take a look on the Python code we wrote in order to perform sentiment analysis on the tweets we gathered by using the operators we described.

4.4 Python Code Review

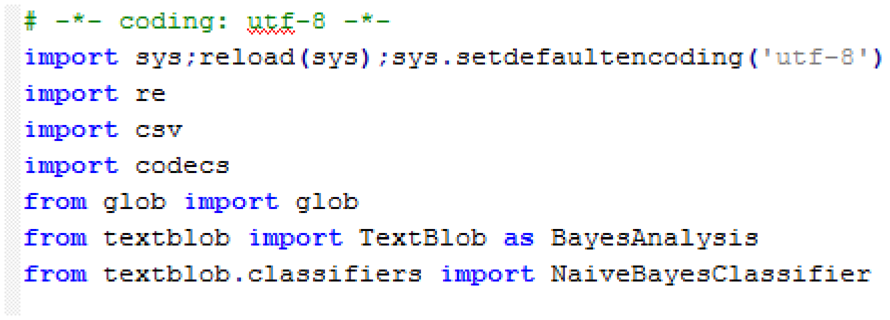
Python is a powerful tool for data analysts around the globe. It offers a variety of libraries and functions that make the process of analysis automated and makes the life of analysts simpler. In our case we used python to train a classifier based on a training corpus that we found online [18] and after the classifier is ready we applied the rules we came up with, on the tweets we downloaded from Twitter’s Search API. Let’s see how we wrote the code that does the sentiment analysis part in Python.

Firstly, we had to add modules and libraries that will help us later in the implementation phase. The first module we imported is the *sys* module. This module provides important information about a variety of functions, methods which are a part of the Python interpreter. There are plenty of extensions that the *sys* module might have. In our case we used sys.setdefaultencoding('utf-8') in order to set as default encoding form the utf-8. This is important because there might be cases that the code breaks when we apply it on differently encoded txt files.

The *re* module that we imported offers operations that match with the regular expressions we will use afterwards. There are many functions that the *re* module includes and can be helpful in sentiment analysis. We can find where exactly a pattern occurs within a string with the *re.search()* function, we can substitute any instances of the pattern we searched for with any string we desire with the *re.sub()* method. To use this method we have to provide the pattern we want to find and also the string with which we want to substitute it with. We can also count and find all the instances of the pattern we provided with *re.findall()* and *re.finditer* functions. Both methods find all occurrences of the pattern but there is a small different in what they do afterwards. The first one, will return a list of the strings in which the pattern is a part of. The second one, will return a list of objects that we can iterate into and get the results we want.

Furthermore, we added the *csv* (Comma Separated Values) module that is one of the most commonly used formats when we have to deal with large databases. Despite the fact that our initial data are downloaded as txt files, we chose to convert the result data in csv files. The *codecs* module we imported helps us define classes in the basic Python codecs both encoders and decoders. It also offers access to the registry of the internal Python codec helping developers deal with codec errors. Also, the *glob* module we imported is based on Unix shell and what it does is to find pathnames that match predefined patterns and return results in arbitrary order.

After importing all the modules we needed to complete the project we started importing useful libraries. A library that we used extensively during this project is textblob. Textblob is a python library that is built to process text files. As we mentioned earlier the initial data files downloaded from the Twitter API are in txt format. What makes this library special is that it provides an API that can deal with requests about natural processing tasks like opinion mining, classification and many more. It is based on NLTK and patterns and can deal with both of them quite nicely. Once we imported the textblob library we also imported the textblob.classifiers module. With this module we can easily build customized classifiers like sentiment analyzers and more. Textblob is a very powerful library and can be used in various data mining projects. For more information about the library you can visit https://textblob.readthedocs.io/en/dev/ [19]. Below we provide an image with the libraries and modules we described earlier:



```
# -*- coding: utf-8 -*-
import sys;reload(sys);sys.setdefaultencoding('utf-8')
import re
import csv
import codecs
from glob import glob
from textblob import TextBlob as BayesAnalysis
from textblob.classifiers import NaiveBayesClassifier
```

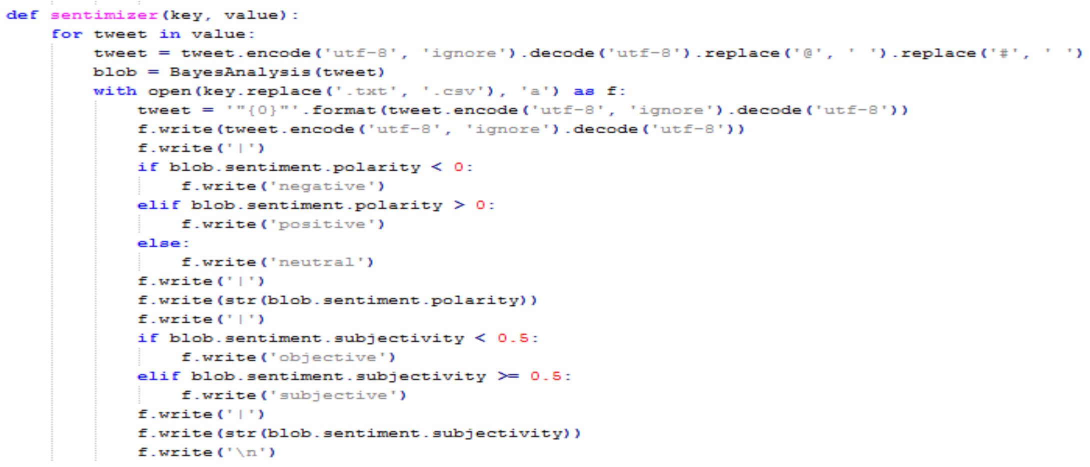
Figure 25: *importing useful libraries and modules in our Python code*

There are 6 modules that we built in order to perform sentiment analysis on our data. Our code is based on Naïve Bayes classification model and what it does is to classify each tweet as positive, negative or neutral depending on their textual context and also set a subjectivity score for each tweet. So we define a polarity and subjectivity score for each tweet. There are three possible values for polarity: negative, positive and neutral. Our code sets a value between -1 and 1 for each tweet (-1 >= *polarity score* >= 1). If the polarity score is 0 the tweet is considered neutral, if the polarity score is less than 0 the tweet is considered negative and if the polarity score is greater than 0 then the tweet is listed as positive. The subjectivity score for each tweet can take values between 0 and 1 (0 >= *subjectivity score* >= 1). Now let’s see how the modules we have on our code do what we described.

The first module that is executed is the *trainer* module. This module is where we train our classifier. In order to do so, the trainer module calls another module that is called *data_builder*. In this module, we open the training dataset and place its tuples in a new list which is named *training_sentences*. Before we place the training dataset in the new list we have to preprocess our training dataset in order to make it readable for TextBlob classifier. To do so, we build a module called *data_parser* which does what we described and also checks the polarity score of each sentence. In our training dataset the sentiment score is quite different from what we want to do in our test data. If a sentence has a polarity score greater or equal to 0.5 it is considered as positive. If the score is less than 0.5 the sentence is considered negative. The *data_parser* module checks what the polarity score is and sets the values of *pos* if the sentence is positive and *neg* if the sentence is negative. After this procedure is over, the training of our classifier (or as we called it NaiveBayesClassifier) begins.

The next module that is executed is the *reader* module. Here is where the sentiment analysis begins. The tweets we downloaded are gathered by calling another module. The *read_from_file* module reads the tweets we downloaded and clears them from unnecessary data like some symbols (*, @ etc.), next it changes the files' extensions making them csv (as we said earlier the downloaded files are in txt format). This module also add two columns in the tweets which are the polarity and subjectivity score. Of course at this point these columns are left blank, after the sentiment analysis process is through these fields will take the scores that the classifier will assign to them.

The next module that will be executed is the *sentimenizer*. Here is where the polarity and subjectivity scores are assigned to every tweet of our database. As we mentioned earlier with the help of *NaiveBayesClassifier*, the *sentimenizer* module checks what score is assigned to each tweet and it classifies them as *positive*, *negative* and *neutral* depending on their polarity score (-1<= *polarity score* <= 1) and as *subjective* and *objective* depending on their subjectivity score (0<= *subjectivity score* >= 1). Below we provide a sample of the code we described and more specifically the *sentimenizer* module.



```python
def sentimizer(key, value):
    for tweet in value:
        tweet = tweet.encode('utf-8', 'ignore').decode('utf-8').replace('@', ' ').replace('#', ' ')
        blob = BayesAnalysis(tweet)
        with open(key.replace('.txt', '.csv'), 'a') as f:
            tweet = '"{0}"'.format(tweet.encode('utf-8', 'ignore').decode('utf-8'))
            f.write(tweet.encode('utf-8', 'ignore').decode('utf-8'))
            f.write('|')
            if blob.sentiment.polarity < 0:
                f.write('negative')
            elif blob.sentiment.polarity > 0:
                f.write('positive')
            else:
                f.write('neutral')
            f.write('|')
            f.write(str(blob.sentiment.polarity))
            f.write('|')
            if blob.sentiment.subjectivity < 0.5:
                f.write('objective')
            elif blob.sentiment.subjectivity >= 0.5:
                f.write('subjective')
            f.write('|')
            f.write(str(blob.sentiment.subjectivity))
            f.write('\n')
```

Figure 26: *the sentimenizer module*

When our python code is executed, csv files are generated. These csv files are the tweets we downloaded classified and ready for our statistical analysis.

4.5 Implementation summary

Now let's make a small summary on what we did so far.

As we mentioned earlier we used PHP in order to retrieve tweets from the Twitter Search API. We changed the hashtags periodically and retrieved a vast amount of data. In the requests we made towards the API we used the same code and the only thing that changed in every request was either the hashtag or the date. We collected data for the following hashtags:

- #DonaldTrump. This is the most common hashtag that Twitter's visitors use to make a statement about the republican candidate Donald J. Trump.
- #HillaryClinton, This is the most common hashtag that Twitter's visitors use to make a statement about the democrat candidate Hillary Clinton.
- #maga. This is the official hashtag in favor of Donald Trump and it's the initials of the republican party's slogan (Make America Great Again)
- #imwithher. This is the official hashtag in favor of Hillary Clinton.
- #NeverTrump. This is the hashtag that is used to show a negative sentiment towards Donald Trump.
- #NeverHillary. This is the hashtag that is used to show a negative sentiment towards Donald Trump.

We collected tweets after the 4 major candidate debates that occurred on September 26th, October 9th, October 4th, October 19th. However, since the data that we gathered were incredibly vast we decided to use only the tweets after the final two debates that were labeled by the media as the most important. The sentiment analysis process we performed only about the tweets that had the following two hashtags "#DonaldTrump" and "#HillaryClinton". We took this decision because when a user choses to include one of the other two hashtags in his/her tweet the sentiment is clear already since they are the official hashtags in favor of the candidates. So, we counted these tweets as positive towards the candidate they represent.

After the data gathering process ended we started designing our classifier. To build this classifier we decided to use the Machine Learning approach. Our target is to classify tweets as positive, negative and neutral in order to see what the voters think about the two candidates. After classifying the tweets we counted the percentage of positive and negative feedback for each candidate and made some assumptions about who is going to win the state that the tweets are from. The states that we decided to gather tweets from are: Florida, Ohio and North Carolina.

As we said we will perform Sentiment Analysis on tweets with the help of Machine Learning and more specifically with Naïve Bayes method. To achieve our goal we implemented our model in small steps in order to make it as accurate and detailed as possible. We will describe the steps our project consists of:

1. We downloaded a training dataset to help us build our model. As we mentioned earlier when performing Machine Learning classification we need a corpus that is already classified in order help us build a classifier. There are many free available corpuses that can be downloaded and be used as training data. The corpus we used as training data is a combination of the Niek Sanders dataset and the corpus that Michigan University developed during a sentiment analysis competition on Kaggle. After getting both the training and the test data we have to preprocess them in order to make them easier for us to handle.
2. Afterwards we preprocessed the tweets in order to make them simpler to process. We got rid of unnecessary information like usernames, dates and hashtags and process only the tweets' text.
3. We the built our classifier based on the training data we downloaded. We set a subjectivity and polarity value for each tweet with the help of textblob. Tweets with polarity value less than 0 are negative, equal to 0 neutral and greater than 0 positive. The subjectivity score indicates if the opinion expressed in the tweet is a subjective or objective. The value of subjectivity is between 0 and 1. The higher the value the more subjective the statement is.
4. We then counted the number of negative and positive tweets in order to make our prediction. Our innovation is that apart from the sentiment analysis process, we collected tweets that contain specific hashtags that reveal sentiment themselves. For example tweets that contain the hashtags #NeverHillary and #NeverTrump don't need to be analyzed to assume that they express negative sentiment. The exact opposite hashtags are #maga and #imwithher which are official hashtags in favor of the candidates.

After we successfully completed every step we described above, the implementation process was finished. We classified tweets from three states as positive, negative and neutral and we also gathered tweets with hashtags that express sentiment without having to analyze them (#NeverTrump, #NeverHillary etc.). Next, we will make a few test cases to check if our code (both in PHP and Python) work properly and finally see if the predictions we made are close to the final results.

5. Testing and Evaluation

5.1 Testing

In this chapter we are going to make a few test cases in order to make sure our code works properly without any malfunctions. Our implementation process has two main parts:

- The PHP code with which we made requests to the API and
- The Python code with which we made the sentiment analysis on the tweets we gathered

In order to make sure these two pieces of code run as planned we are going to check if the execution process is done correctly. We will start by testing our PHP code.

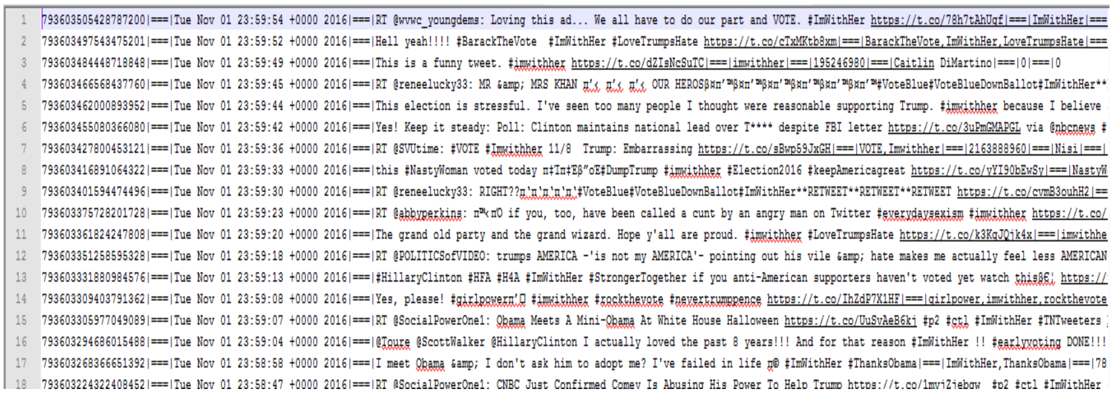
5.1.1 Testing our PHP code

Since the PHP we wrote establishes a connection with the API we have to make sure that this connection is persistent and stable. As the data we gathered show, we made 31 requests in total. This means that we retrieved 31 txt files that contained tweets from the three states we mentioned earlier. However, before we started gathering tweets we tested if our code runs properly by making random requests for random topics. We made 20 requests before starting the gathering process. In these requests we draw the conclusion that the code works exactly as planned.

The most important thing for us was to check if there is a case where the code crashes while collecting tweets. After 51 test we run, we can say that we didn’t witness any crashes while executing the script. What is really interesting, however, is that when we set a date older than 7 days before the current date the code didn’t run. For example, let’s say that the current date is 20/12/2016. If we run the code asking for tweets older than 13/12/2016, let’s say 9/12/2016 the code will not be executed. This doesn’t mean that the code has a problem. We said earlier that the Search API has some limits. We can only get tweets that are not older than one week. Consequently, we can’t make requests in order to retrieve old tweets with this API. In conclusion, besides this issue with the API the code run properly (if we set the correct date) every time it was executed.

As we mentioned, we built the code in such a way that during the execution we can see how many tweets are gathered so far. One of the most challenging tasks was to deal with the paging issue we described earlier. To do so, we set a limit for each page to have no more than 100 tweets. So if we made a request that will get us 9000 tweets we would get approximately 90 pages with tweets each of whom, wouldn’t have more than 100 tweets. In the 51 requests we made the code responded correctly placing the tweets in pages without exceeding the threshold we set for each page. As we said, the whole process is shown in the command line during the execution of the code.

After the execution of the code a txt file is generated containing tweets in text format. We need to check if the tweets are gathered correctly and if the parameters we defined are met successfully. In fact, each tweet we gathered has all the fields we want it to have separated with the following symbol "|===|". In addition, the parameters we set are met successfully in every tweet we gathered: all tweets are in English, the dates we defined to gather tweets from are also correct. Below we provide an instance of the tweets we downloaded to show that the files are exactly how we wanted them to be.



```
793603505428787200|===|Tue Nov 01 23:59:54 +0000 2016|===|RT @wvwc_youngdems: Loving this ad... We all have to do our part and VOTE. #ImWithHer https://t.co/78h7tAhUqf|===|ImWithHer|===
793603497543475201|===|Tue Nov 01 23:59:52 +0000 2016|===|Hell yeah!!!! #BarackTheVote  #ImWithHer #LoveTrumpsHate https://t.co/cTxMKtb8xm|===|BarackTheVote,ImWithHer,LoveTrumpsHate|===
793603484448718848|===|Tue Nov 01 23:59:49 +0000 2016|===|This is a funny tweet. #imwithher https://t.co/dZIsNcSuTC|===|imwithher|===|195246980|===|Caitlin DiMartino|===|0|===|0
793603466568437760|===|Tue Nov 01 23:59:45 +0000 2016|===|RT @reneelucky33: MR &amp; MRS KHAN п'‹ п'‹ п'‹ OUR HEROSβ¤п'™β¤п'™β¤п'™β¤п'™β¤п'™#VoteBlue#VoteBlueDownBallot#ImWithHer**
793603462000893952|===|Tue Nov 01 23:59:44 +0000 2016|===|This election is stressful. I've seen too many people I thought were reasonable supporting Trump. #imwithher because I believe
793603455080366080|===|Tue Nov 01 23:59:42 +0000 2016|===|Yes! Keep it steady: Poll: Clinton maintains national lead over T**** despite FBI letter https://t.co/3uPmGMAPGL via @nbcnews #
793603427800453121|===|Tue Nov 01 23:59:36 +0000 2016|===|RT @SVUtime: #VOTE #Imwithher 11/8  Trump: Embarrassing https://t.co/sBwp59JxGH|===|VOTE,Imwithher|===|2163888960|===|Nisi|===|
793603416891064322|===|Tue Nov 01 23:59:33 +0000 2016|===|this #NastyWoman voted today п‡Iп‡Eβ"oE#DumpTrump #imwithher #Election2016 #keepAmericagreat https://t.co/yYI90bEwSy|===|NastyW
793603401594474496|===|Tue Nov 01 23:59:30 +0000 2016|===|RT @reneelucky33: RIGHT??п'п'п'п'п'#VoteBlue#VoteBlueDownBallot#ImWithHer**RETWEET**RETWEET**RETWEET https://t.co/cvmB3ouhH2|==
793603375728201728|===|Tue Nov 01 23:59:23 +0000 2016|===|RT @abbyperkins: п™‹п0 if you, too, have been called a cunt by an angry man on Twitter #everydaysexism #imwithher https://t.co/
793603361824247808|===|Tue Nov 01 23:59:20 +0000 2016|===|The grand old party and the grand wizard. Hope y'all are proud. #imwithher #LoveTrumpsHate https://t.co/k3KqJQjk4x|===|imwithhe
793603351258595328|===|Tue Nov 01 23:59:18 +0000 2016|===|RT @POLITICSofVIDEO: trumps AMERICA -'is not my AMERICA'- pointing out his vile &amp; hate makes me actually feel less AMERICAN
793603331880984576|===|Tue Nov 01 23:59:13 +0000 2016|===|#HillaryClinton #HFA #H4A #ImWithHer #StrongerTogether if you anti-American supporters haven't voted yet watch thisβ€¦ https://
793603309403791362|===|Tue Nov 01 23:59:08 +0000 2016|===|Yes, please! #girlpowerп'□ #imwithher #rockthevote #nevertrumppence https://t.co/IhZdP7X1HF|===|girlpower,imwithher,rockthevote
793603305977049089|===|Tue Nov 01 23:59:07 +0000 2016|===|RT @SocialPowerOne1: Obama Meets A Mini-Obama At White House Halloween https://t.co/UuSvAeB6kj #p2 #ctl #ImWithHer #TNTweeters
793603294686015488|===|Tue Nov 01 23:59:04 +0000 2016|===|@Toure @ScottWalker @HillaryClinton I actually loved the past 8 years!!! And for that reason #ImWithHer !! #earlyvoting DONE!!!
793603268366651392|===|Tue Nov 01 23:58:58 +0000 2016|===|I meet Obama &amp; I don't ask him to adopt me? I've failed in life п© #ImWithHer #ThanksObama|===|ImWithHer,ThanksObama|===|78
793603224322408452|===|Tue Nov 01 23:58:47 +0000 2016|===|RT @SocialPowerOne1: CNBC Just Confirmed Comey Is Abusing His Power To Help Trump https://t.co/1mvjZiebgw  #p2 #ctl #ImWithHer
```

Figure 27: *instance of tweets we gathered*

To conclude our testing for the PHP code, we can say that the code runs correctly 100% of the times we tested it. The only issue comes from the API, but it didn't affect our data gathering process. The code can be executed in any machine (we tested it in three different systems) and the results are exactly what we want. We encourage anyone to test it by following the instructions we mentioned.

5.1.2 Testing Python code

The second part of the implementation process for our project is the Python code. This script trains a classifier based on a training dataset and afterwards performs sentiment analysis on the tweets we gathered. This process is extremely complicated and time consuming since we have to deal with a total of 277.509 tweets that need to be classified separately. To test if everything works fine with our code we are going to test the following cases:

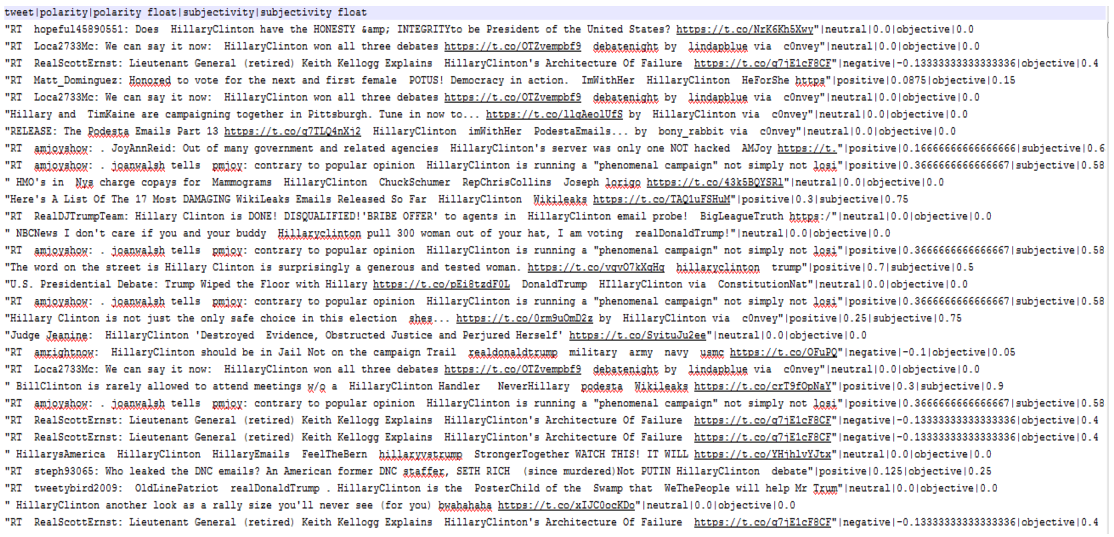
- Execute the code multiple times to check its consistency
- Take a sample of tweets to check if they are correctly classified as positive, negative, neutral (based on the polarity score) and also as subjective and objective (based on the subjectivity score).
- Check if the result csv files are structured as planned

In the first case, we can say that we executed the script more than 10 times to see if the whole process is done properly. The code is built under the premise that we have three different files each one named after the state the tweets are from and they contain the tweets from each state separately. Furthermore, the code indicates a path that the training dataset is in, in order to train the classifier. These are the paths that need to be changed in order to run the code (depending on the system). We executed the code in three different systems and it run properly in every one of them.

A very common issue when performing sentiment analysis on social media, is that a lot of users choose sarcasm in order to express themselves. While searching through the result files we detected that a really small percentage of the tweets we fetched were misclassified. That happened because for example there were some tweets that even though used words with positive sentiment they were actually sarcastic. We took a sample of 1000 tweets and checked them manually one by one to check what is the percentage of misclassified tweets. From the 1000 tweets we checked, 67 tweets were misclassified. So we have a percentage of 6.7% misclassified tweets. This percentage is very satisfying since we are talking social media and different people that express themselves with different perspectives.

On the other hand, the subjectivity score has a better percentage of accuracy. This is natural since it is easier to detect the subjective words and count them, than determine which of these words are positive and negative. From the 1000 sample tweets we studied, all of them where correctly classified as objective or subjective. Regardless the fact we didn’t use the subjectivity score in our final results, the scores are available for future researchers to exploit them in any way they find more suitable.

After the sentiment analysis process is through, the script produces new csv files. These files are the old txt files, classified and altered as the script indicates. Let’s see if the new csv files are structured as we wanted. We decided that the new csv files should contain only the username, the tweet’s text, the polarity, the polarity score, the subjectivity and the subjectivity score. We can say that all tweets have in fact these fields as the sample image below shows:



```
tweet|polarity|polarity float|subjectivity|subjectivity float
"RT  hopeful45890551: Does  HillaryClinton have the HONESTY &amp; INTEGRITYto be President of the United States? https://t.co/NrK6Kh5Xwy"|neutral|0.0|objective|0.0
"RT  Loca2733Mc: We can say it now:  HillaryClinton won all three debates https://t.co/OTZvempbf9  debatenight by  lindapblue via  c0nvey"|neutral|0.0|objective|0.0
"RT  RealScottErnst: Lieutenant General (retired) Keith Kellogg Explains  HillaryClinton's Architecture Of Failure  https://t.co/q7jE1cF8CF"|negative|-0.13333333333333336|objective|0.4
"RT  Matt_Dominguez: Honored to vote for the next and first female  POTUS! Democracy in action.  ImWithHer  HillaryClinton  HeForShe https"|positive|0.0875|objective|0.15
"RT  Loca2733Mc: We can say it now:  HillaryClinton won all three debates https://t.co/OTZvempbf9  debatenight by  lindapblue via  c0nvey"|neutral|0.0|objective|0.0
"Hillary and  TimKaine are campaigning together in Pittsburgh. Tune in now to... https://t.co/llqAeolUfS by  HillaryClinton via  c0nvey"|neutral|0.0|objective|0.0
"RELEASE: The Podesta Emails Part 13 https://t.co/q7TLQ4nXj2  HillaryClinton  imWithHer  PodestaEmails... by  bony_rabbit via  c0nvey"|neutral|0.0|objective|0.0
"RT  amjoyshow: . JoyAnnReid: Out of many government and related agencies  HillaryClinton's server was only one NOT hacked  AMJoy https://t."|positive|0.16666666666666666|subjective|0.6
"RT  amjoyshow: . joanwalsh tells  pmjoy: contrary to popular opinion  HillaryClinton is running a "phenomenal campaign" not simply not losi"|positive|0.3666666666666667|subjective|0.58
" HMO's in  Nys charge copays for  Mammograms  HillaryClinton  ChuckSchumer  RepChrisCollins  Joseph lorigo https://t.co/43k5BQYSR1"|neutral|0.0|objective|0.0
"Here's A List Of The 17 Most DAMAGING WikiLeaks Emails Released So Far  HillaryClinton  Wikileaks https://t.co/TAQ1uFSHuM"|positive|0.3|subjective|0.75
"RT  RealDJTrumpTeam: Hillary Clinton is DONE! DISQUALIFIED!'BRIBE OFFER' to agents in  HillaryClinton email probe!  BigLeagueTruth https:/"|neutral|0.0|objective|0.0
" NBCNews I don't care if you and your buddy  Hillaryclinton pull 300 woman out of your hat, I am voting  realDonaldTrump!"|neutral|0.0|objective|0.0
"RT  amjoyshow: . joanwalsh tells  pmjoy: contrary to popular opinion  HillaryClinton is running a "phenomenal campaign" not simply not losi"|positive|0.3666666666666667|subjective|0.58
"The word on the street is Hillary Clinton is surprisingly a generous and tested woman. https://t.co/vqvO7kXqHq  hillaryclinton  trump"|positive|0.7|subjective|0.5
"U.S. Presidential Debate: Trump Wiped the Floor with Hillary https://t.co/pEi8tzdF0L  DonaldTrump  HIllaryClinton via  ConstitutionNat"|neutral|0.0|objective|0.0
"RT  amjoyshow: . joanwalsh tells  pmjoy: contrary to popular opinion  HillaryClinton is running a "phenomenal campaign" not simply not losi"|positive|0.3666666666666667|subjective|0.58
"Hillary Clinton is not just the only safe choice in this election  shes... https://t.co/0rm9uOmD2z by  HillaryClinton via  c0nvey"|positive|0.25|subjective|0.75
"Judge Jeanine:  HillaryClinton 'Destroyed  Evidence, Obstructed Justice and Perjured Herself' https://t.co/SvituJu2ee"|neutral|0.0|objective|0.0
"RT  amrightnow:  HillaryClinton should be in Jail Not on the campaign Trail  realdonaldtrump  military  army  navy  usmc https://t.co/OFuPQ"|negative|-0.1|objective|0.05
"RT  Loca2733Mc: We can say it now:  HillaryClinton won all three debates https://t.co/OTZvempbf9  debatenight by  lindapblue via  c0nvey"|neutral|0.0|objective|0.0
" BillClinton is rarely allowed to attend meetings w/o a  HillaryClinton Handler   NeverHillary  podesta  Wikileaks https://t.co/crT9fOpNaY"|positive|0.3|subjective|0.9
"RT  amjoyshow: . joanwalsh tells  pmjoy: contrary to popular opinion  HillaryClinton is running a "phenomenal campaign" not simply not losi"|positive|0.3666666666666667|subjective|0.58
"RT  RealScottErnst: Lieutenant General (retired) Keith Kellogg Explains  HillaryClinton's Architecture Of Failure  https://t.co/q7jE1cF8CF"|negative|-0.13333333333333336|objective|0.4
"RT  RealScottErnst: Lieutenant General (retired) Keith Kellogg Explains  HillaryClinton's Architecture Of Failure  https://t.co/q7jE1cF8CF"|negative|-0.13333333333333336|objective|0.4
" HillarysAmerica  HillaryClinton  HillaryEmails  FeelTheBern  hillaryvstrump  StrongerTogether WATCH THIS! IT WILL https://t.co/YHjhlvYJtx"|neutral|0.0|objective|0.0
"RT  steph93065: Who leaked the DNC emails? An American former DNC staffer, SETH RICH  (since murdered)Not PUTIN HillaryClinton  debate"|positive|0.125|objective|0.25
"RT  tweetybird2009:  OldLinePatriot  realDonaldTrump . HillaryClinton is the  PosterChild of the  Swamp that  WeThePeople will help Mr Trum"|neutral|0.0|objective|0.0
" HillaryClinton another look as a rally size you'll never see (for you) bwahahaha https://t.co/xIJC0ocKDo"|neutral|0.0|objective|0.0
"RT  RealScottErnst: Lieutenant General (retired) Keith Kellogg Explains  HillaryClinton's Architecture Of Failure  https://t.co/q7jE1cF8CF"|negative|-0.13333333333333336|objective|0.4
```

Figure 28: *instance of classified tweets*

5.2 Evaluation of the project

As we discussed earlier, both PHP and Python scripts work properly. But this isn’t enough for a project that is trying to predict an event like the presidential elections in

USA. To evaluate our project we need to compare the predictions we made with the actual results of the elections.

This project tried to make a prediction about the USA elections in three key states that from our point of view, would determine the final results. Let’s see what our sentiment analysis predicted and how close these predictions are to the actual election results.

5.2.1 Florida results

73263 tweets were gathered from the state of Florida. From these 73263 tweets 42683 were in favor of Donald Trump, 25938 were in favor of Hillary Clinton and 4642 tweets were classified as neutral. To sum up the results that our analysis predicted were:

- Donald Trump: 58.26%
- Hillary Clinton: 35.4%
- Neutral tweets: 6.44%

Now let’s see the actual results of the elections in the state of Florida (99% of votes counted so far):

- Donald Trump: 49.1%
- Hillary Clinton: 47.8%
- Other Candidates: 2.9%

We can see that the percentages are pretty close, but the most important thing is that we correctly predicted who would win the state.

5.2.2 Ohio results

116733 tweets were gathered from the state of Ohio. From these tweets 65.611 tweets were in favor of Donald Trump, 41727 tweets were in favor of Hillary Clinton and 9395 were classified as neutral. So, the percentages that the analysis gave us are:

- Donald Trump: 56.2%
- Hillary Clinton: 35.75%
- Neutral tweets: 8.05%

Below we can see what the actual results are (94% of votes counted so far):

- Donald Trump: 52.1%
- Hillary Clinton: 43.5%
- Other Candidates: 4%

Similar to Florida, we can observe that the results are close but the important thing is that once again we predicted who the final winner will be.

5.2.3 N. Carolina results

The final state we gathered tweets from is N. Carolina. 87513 tweets were gathered in total from this state. From these 87513 tweets, 46608 were in favor of Donald Trump, 33301 were in favor of Hillary Clinton and 7604 tweets were classified as neutral. So, the percentages that our analysis predicted are:

- Donald Trump: 53.26%
- Hillary Clinton: 38.05%
- Neutral tweets: 8.69%

The actual results of the state are (98% of votes counted so far):

- Donald Trump: 50.5%
- Hillary Clinton: 46.7%
- Other Candidates: 2.8%

Once again the results are really close and the most important thing is that we correctly predicted who would win the state again

Final Evaluation of the project

The charts below show the results of our analysis for all three states:

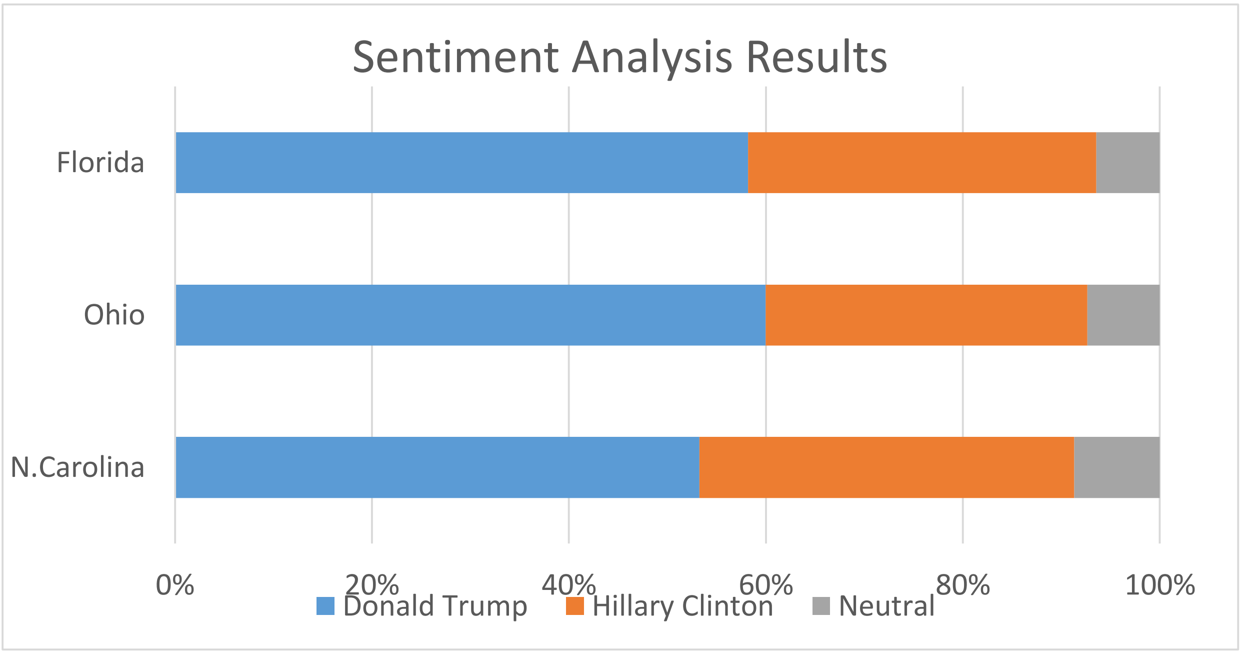


Figure 29: *Sentiment analysis results*

The chart below shows the actual results of the elections:

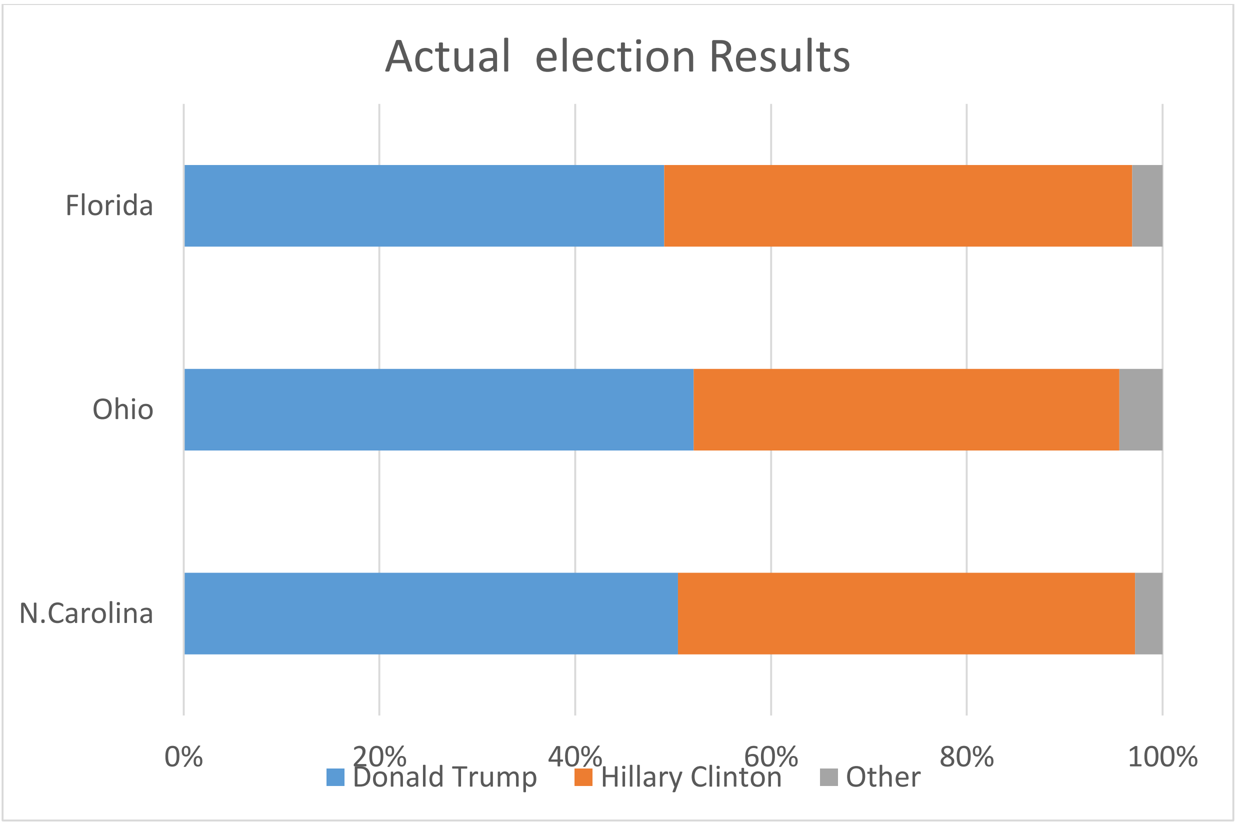
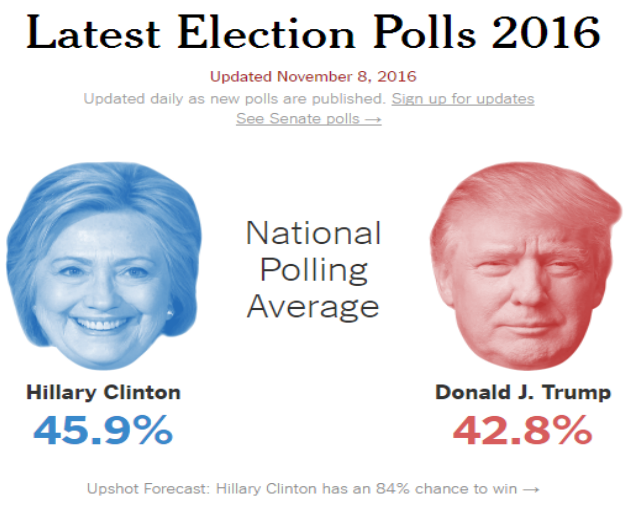


Figure 30: *Actual elections' results*

We can see that the two charts are very similar. In a race where the two candidates where really close to each other our project correctly predicted the results with high accuracy. The most impressive thing here is that every poll we found was indicating that Hillary Clinton would win the elections. Below we give some polls of some well-known websites and channels to show what everyone else was predicting before the elections.

On the day of the elections New York Times [20] had the following post on their homepage:


Latest Election Polls 2016

Updated November 8, 2016

Updated daily as new polls are published. Sign up for updates

See Senate polls →

National Polling Average

Hillary Clinton 45.9%

Donald J. Trump 42.8%

Upshot Forecast: Hillary Clinton has an 84% chance to win →

Figure 31: *Poll taken from the New York Times website on November 7th* [20]

As we can see Hillary Clinton is ahead. USA today [21] uploaded the following poll on the elections day:

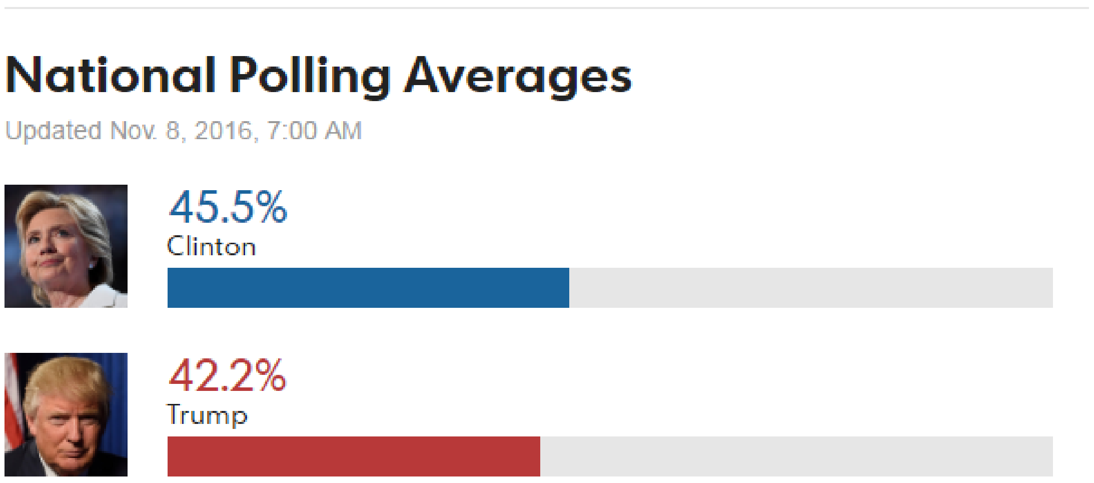
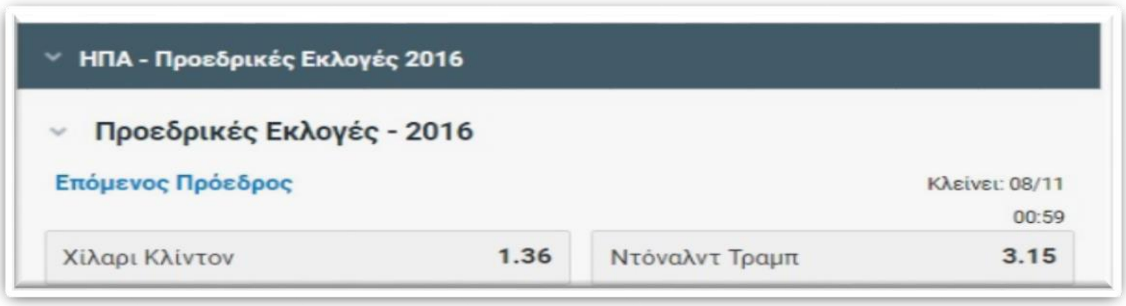


Figure 32: *USA today predictions on November 7th* [21]

Once again Hillary Clinton is ahead. Finally let’s see the odds that a Greek betting site [22] offered to whoever wanted to bet on the elections:



ΗΠΑ - Προεδρικές Εκλογές 2016

Προεδρικές Εκλογές - 2016

Επόμενος Πρόεδρος

Κλείνει: 08/11 00:59

Χίλαρι Κλίντον	1.36	Ντόναλντ Τραμπ	3.15

Figure 33: *Greek betting site odds for the USA presidential elections*

Hillary Clinton is again considered as the most probable next President.

We provide these predictions to show that this project predicted something that was very unlikely to happen. Even though we made a research based on social media and not on polls and votes the results were extremely accurate and this shows that the project fulfilled its purpose and can be considered as successful. Now let’s see the conclusions we made, both technical and theoretical and what can be done in the future to enhance this project further.

6. Conclusions and future work

6.1 Conclusions

After finishing with the implementation process and submitting our predictions for the elections that occurred on November 8th 2016, we can say that our experiment was successful. We predicted correctly who will win in all three states we gathered tweets from. Consequently, we can say that Twitter is a social media that analysts can use in order to draw useful conclusions about elections but also for other things as well like products, events, market issues and so on.

Of course, we can’t claim that analyzing tweets is the ideal way of predicting election results with high accuracy since a lot of voters don’t have a Twitter account, but we can see what the public opinion says about different topics and make some low risk assumptions about the future. In this dissertation we try to observe how the voters feel about the candidates, we do not count how many votes each one will get. Thus, we cannot directly predict the winner. However, the statistical analysis and polls that we cited in the previous chapter show that predicting the winner of such an event is not a simple task. Such problems should be addressed in multidimensional ways and take many factors into consideration before making a prediction.

In our case what was really odd is that even though all polls showed that Hillary Clinton would eventually win the elections, Twitter users mainly mentioned Donald Trump when they posted a tweet about the elections. As a result, when we made a request with the Search API to retrieve tweets for Donald Trump using hashtags like #DonaldTrump, #maga or the negative #NeverTrump we ended up with a lot more tweets than when we made a request for Hillary Clinton. So we can say that people were talking about Donald Trump a lot more than they talked about Hillary Clinton. It doesn’t matter that a large percentage of these tweets were negative for Trump. It seems that there is no such thing like negative publicity.

In the technical point of view, we have to say that we were extremely impressed with the variety of options that the Twitter offers to download data. More specifically, the Search API is built in a way that developers can use it easily and make the requests they want to get their data. For analysts that want a different approach Twitter offers the Streaming API which offers a real-time perspective and can be used easily as well. In general, for analysts that want to gather data from social media we recommend that they do it with Twitter.

Furthermore, we conducted the result that when we are dealing with APIs the best solution is PHP. We made a few tries with JAVA but it turned out that practically it’s not a good idea. PHP is extremely easy to learn. It’s the first time we wrote code in PHP and we can say that the documentation and the forums offered are extremely helpful for beginners and when a problem occurs it is almost sure that the solution is available in the PHP community.

Finally, we can say that Python is the best tool for analysts that want to perform any kind of analysis on data. It offers a variety of libraries and modules that are designed for analytical purposes and that helps developers to save a lot of time.

6.2 Future Work

6.2.1 Different approaches

This dissertation shows that we should see social media as an opportunity to gather and analyze data in order to make predictions and assumptions about the future. We established that social media can, in fact, be used to draw conclusions about serious events like elections. However, there are some things that can be done in future researches that might get this project on another level, making it even more accurate than it already is.

Firstly, the sentiment analysis process can be done by seeing the whole project from another perspective. We used machine learning's Naïve Bayes model to do it on this project, but in future projects the support vector machine approach can be used as well. They might fall under the same category of machine learning, but there are some differences between them that make it intriguing to also test the second one also. There are cases that SVM might perform better than Naïve Bayes model if we choose to base our analyzation on other parameters.

In addition, we can also try to address this problem using the Rule Based approach. This approach is not effective for unstructured data, but in our case the data is structured so this method might be applied with success. We have to make as many tests as possible to determine which is the best way to approach the problem. We will try to briefly see this problem from the Rule Based point of view without proceeding with the implementation.

Now let's see some examples that describe how the Rule Based approach works. First, we have to see the text we will deal with and classify all the subjective words as positive or negative. For instance, words like "love", "like", "admire" are considered positive and the text that contains them is probably a positive feedback about the subject. On the other hand, words like "hate", "dislike" have a negative meaning so the text that contains them is most probably a negative review. Now there might be a case that a text contains both positive and negative words. In that case we have to count the total of positive and negative words and determine the polarity of the text according to what words prevail in total. What we just described is considered one of the simplest Rule Based approach. It is really difficult and time consuming to classify words as positive or negative. There are millions of words that have to be classified and that is something most analysts try to avoid. There are ways of dealing with this problem without investing a lot of time. To do this we will have to use lexicons. A lexicon is a resource that contains information about words. A very simple example of a lexicon is a dictionary. In our case we need a lexicon that has already classified the words it contains as negative or positive. There are a lot of hand annotated lexicons available for downloading from well-known universities and research papers. Those lexicons make the opinion mining process a lot simpler.

Let's some examples of tweets in text format to understand exactly what this approach is about. "I really like what Hillary says about women rights. She is the best choice for the upcoming elections", this tweet has a clear positive vibe about Hillary Clinton. The key words in this case are "really like" and "best choice". Those two phrases make this tweet a positive feedback. "Seriously, I believe everyone really hates Hillary. I think the email scandal was a disaster", in this case we have a negative review. The key words here are "really hates" and "disaster". Those two phrases make the tweet negative.

However, in most cases people are not so straight forward when expressing their opinions in social media. Most of them use very complex mechanisms like irony and sarcasm which makes the whole process extremely challenging for analysts. In addition, there are cases that even though positive words prevail in a text document, the actual meaning of the text is negative. There is a famous letter that someone wrote that analysts use to demonstrate how a text may seem to have a positive vibe but it actually doesn't. The letter was from a customer that praised the competitor company saying why their product is better and at the end of the letter he ended with the phrase "you suck". So, even though the letter was full with positive words (about the competitor) it shouldn't be considered as positive. This is the most difficult challenge in sentiment analysis.

A very important aspect of the whole idea behind opinion mining is words that full under the category of intensifiers. These words are usually adverbs that emphasize what the individual is saying. For example the word "really" before the word "hate" or "love" gives a higher intensity in the phrase. So we can say that there are phrases that are "more positive or negative" than others when curtain words are used. In our project we will take into account intensifiers as well.

There are a lot of models that are based in the rule based approach. One of the most efficient ones is the VADER model. This model takes into account many different rules and it combines them in order to come up with conclusions. Here are some example of rules that VADER researchers came up with while designing the model. They included CAPS, exclamation points and emoticons in their vocabulary. Future researchers that want to enhance this project can use these rules. This is important because most people that use social media also use these methods to express their opinion about a topic. For example, "Donald Trump is AWSOME!!! ;)". Here we can see the word awesome in written in caps to emphasize the meaning. Also the three exclamation points also is way to highlight what you're saying. Finally the smiley emoticon at the end of the text also reveals a positive vibe towards Donald Trump. There are also some words that show a shift in emotions. Words like "but" and "however" show a change of the meaning in the text. "I really liked Donald Trump at first but after the tax scandal I changed my mind”. This sentence is clearly a negative review but if we count the positive words we are going to have to classify it as positive. The VADER model helps us deal with these kind of problems.

Now, the problem with rule based approaches is that they are more suitable for large texts. In Twitter each user has a limit in words. So, we mostly have short messages that need to be processed in order to determine their polarity. This is why we decided to use machine learning to address this problem. However, future researchers can try it in order to prove or doubt this theory.

Another issue that can be issued in future work is the amount of data gathered. We strongly believe that the predictions we made would be even more accurate if we gathered more data. The lack of time forced us to gather 277509 tweets in total. The number is quite big but future researchers can collect even more data and achieve even greater accuracy.

Now let’s see some tools we mentioned earlier that can be used to enhance this project in the future. We will start with Lexicons.

6.2.2 Lexicons

Lexicons are very helpful when performing sentiment analysis. They are useful resources of information about vocabulary. A very simple example of a Lexicon is a dictionary. In our case we need a lexicon that defines words as positive or negative. The intensity of a word is also really important. There are words that express feelings more strongly than others. For example the word “terrible” expresses a higher level of dissatisfaction than the word “bad”. This is helpful regardless of what approach we have chosen to go with (Machine Learning or Rule Based).

It’s not difficult to find lexicons nowadays. There are a lot of researchers and universities that design lexicons and make them available for everyone to download. The most commonly used free available lexicons are the following:

- Inquirer
- MPQA
- LIWC
- SentiWordNet

Inquirer, MPQA and LIWC are based on the same idea. They provide categories of words like negative, positive, strong, weak, subjective, objective and so on. A word can be listed in more than one categories. SentiWordNet is pretty different. It is based on another lexicon called WordNet. It provides categories as well but in enhances them with scores of various emotions. Of course there are plenty of other lexicons as well but for this project we encourage future students to use this particular lexicon.

SentiWordNet Lexicon

Now we will describe briefly how this lexicon works and what features it includes. As we discussed in the previous paragraphs SentiWordNet is based on the WordNet lexicon, which is a very special type of lexicon. It is considered to be a network of relationship between words. When we mention relationship between words we don't mean words that look alike, but words that have similar meaning. We are going to see how the WordNet lexicon is constructed because SentiWordNet is based on the same principles.

Every word in the English dictionary can be used in different ways. There are some words that might have different meanings in every sentence they are used. They may be used as an adverb in some cases and as a noun in some others. Let's see an example of sentences where a word might have different meanings. For example "I like cats" and "This person is treating like a baby" both have the word "like" in them. However, the meaning of the word is different in the two sentences. In the first sentence "like" is a verb. In the second sentence it is used as a preposition. So, in the first instance like is a verb showing preference and in the second like is a preposition and shows similarity. This means that we have pairs of words. Verb – Preference and Preposition – similarity. In WordNet each and every one of these pairs is considered an entity and is called a lemma. The two lemmas we have in this case is Like – (Verb - Preferences) and Like – (Preposition - Similarity).

The most important term that we have to understand here is the word lemma. Each lemma is represented with a unique code in the lexicon and it includes three things:

1. The word itself
2. The part of the speech the word is (preposition, adjective, adverb etc)
3. The meaning of the word

Now let's assume that there is a word that is used in two sentences as an adverb but its meaning is different in the two first sentences. Despite the fact that 1 and 2 are satisfied the third part of the lemma which is the meaning of the word is different. Consequently, we will come up with two different lemmas with different and unique codes. This is very important as we will see later.

A very important task that the WordNet lexicon does also is the synset operation. A synset is a group of lemmas that have the same meaning. So, if there are different words that have similar meaning they will be grouped together as a synset. In addition, WordNet provides relationships between synsets. For example, Ferrari is a type of car. WordNet defines their relationship as Type-Of (Ferrari is a type of car). There are many relationships that WordNet offers and we will see in the upcoming chapters.

SentiWordNet operates in the same way as WordNet, but with a slight difference. In particular, it assigns a polarity score for every synset. It is easily understandable that SentiWordNet doesn't directly classify words as negative or positive like the other lexicons do, but defines a polarity and intensity of sentiment for each word. The scores that it assigns for every lemma are:

- Positive polarity score
- Negative polarity score
- Objectivity score

So, every lemma has a polarity score. Let's see an example of a lemma. As we said every lemma has a different code in order to be uniquely identified. The lemma love.v.01 for example is the word love is a verb and its meaning is defined "affections feelings for something or someone". Now the polarity scores are the following:

- Positive polarity score, 0.900
- Negative polarity score, 0
- Objectivity score, 0.100

This means that the synset love.v.01 is used in a positive sense 90% of the time and in an objective sense 10% of the time. 0% of the time love.v.01 is used to show negative feelings. As we can see the total of the score is 1. That happens for every lemma. The score must always add up to 1.

We described with high accuracy how the SentiWordNet lexicon works. In future project this lexicon can play an important role for sentiment analysis.

6.2.3 Other tools for sentiment analysis

The code we wrote was on Anaconda. This is a Python IDE that offers a lot of advantages. We strongly encourage students that want to enhance this project t use it as well. Its major advantage is the ability it gives to developers to simplify the package management process. Of course it is powered by Python itself. This is very important because IDEs that are not developed by the language itself are most probably inadequate.

A very interesting module that we didn't use in our project is PIP. PIP is a tool that helps in package management that is used to install and handle software packages that are important for projects. The available packages for python can be found on Python Package Index. With PIP we can also install libraries and modules that other developers built in order to make our project's implementation simpler. What makes PIP special is that after downloading and installing the libraries and modules we need, it helps us adjust them to our needs and more specific to our project. In addition, the simplicity of installing a library with PIP is remarkable. All we have to do is write a simple command and the library is installed and ready to use. The piece of code that is required in order to install a library with PIP is the following:

```
!PIP install XLSXWRITER
```

With this line the library XLSXWRITTER is installed. It's that simple.

A very important challenge when coding with python is to choose the libraries that will make our life easier and simpler. In projects like ours analysts have to deal with big arrays since the datasets are very big. The problem of the large volume of data can be dealt with two libraries that enable mathematic operations with the array elements. Python lists are great in general, but in this project it's better to put our data to grids and also perform some operations, mostly mathematical, within each element. There are two libraries that are built for these kinds of operations. Their names are numpy and scipy. The first one is a library that helps us manipulate data and adjust it depending on the approach chosen. A numpy array is a little different than a python list since it gives the ability to the user to perform complex mathematical operations like square root, power of etc. Furthermore with numpy we have the ability to find distances between the arrays we have. This very important for projects like ours.

A numpy array is just a set of points. Points might have one dimension or they can be multidimensional. Points with one dimension are represented by a simple number. If we have a point with let's say two dimensions, it will be represented with two numbers and so on. The number of dimensions that the elements of the array have define the rank of the array. There are plenty of attributes that a numpy array has and we have to be familiar before coding. The first attribute we are going to describe is the shape of the array. By shape, we mean the number of rows and columns of the array. Also, indexing is very important when dealing with datasets. We can index the elements of the array just like we do with python lists. Numpy offers a simplistic method of indexing and it is very helpful when we navigate in a dataset. There are a lot of ways to index a numpy array. This gives us a variety of options and flexibility.

The other library we mentioned is scipy. This particular library helps us make complex mathematical operations. These operations might include the compute of the distance between two instances of arrays. The module that is the most useful here is the spatial module. This module helps us find the distance between instances. Distance has a lot of meanings in our case and scipy will help compute everything we need. Examples of distances are: Euclidean distance, similarity between entries, correlation and so on. A very interesting module that we recommend future researchers to use in a regular basis as well is the *pdist* that calculates distance between instances of an array.

Everything we mentioned is highly recommended for the future work of this project. This field has a lot to offer to data mining and the more approaches we test the better our results will be. As a last suggestion, we recommend that the subjectivity score we assigned to the tweets should be used to make the sentiment analysis even more sophisticated. We didn't use it because of lack in time, but it's there for everyone that wants to take this project to another level.

7. References and Bibliography

[1] http://www.politico.com/2016-election/results/map/president

[2] Paper: Tweets mining for French Presidential Election

[3] Paper: Mining Twitter Big Data to Predict 2013 Pakistan Election Winner

[4] Paper: Twitter as a tool for predicting election results

[5] Paper: Prediction of Election Result by Enhanced Sentiment Analysis on Twitter Data using Word Sense Disambiguation

[6] Paper: Emotion analysis of Twitter using Opinion mining

[7] Paper: Opinion Mining about a Product by Analyzing Public Tweets in Twitter
[8] https://dev.twitter.com/oauth
[9] https://dev.twitter.com/rest/public/rate-limits
[10] https://twitter.com/search-home

[11] https://twitter.com/search-advanced
[12] https://dev.twitter.com/rest/public/search

[13] https://dev.twitter.com/rest/public/timelines

[14] http://php.net/manual/en/book.curl.php

[15] https://github.com/J7mbo

[16] https://www.daftlogic.com/projects-google-maps-area-calculator-tool.htm

[17] https://www.google.gr/maps

[18] http://thinknook.com/twitter-sentiment-analysis-training-corpus-dataset-2012-09-22/

[19] https://textblob.readthedocs.io/en/dev/

[20] http://www.nytimes.com/interactive/2016/us/elections/polls.html

[21] http://www.usatoday.com/pages/interactives/2016/election/poll-tracker/

[22] http://www.stoiximan.gr