Twitter Data Stream Sentiment Analysis

By

Group 5

Minor project Software Design and Application 2017-2018

Tim Al 4439848  
Casper Spronk 4369475  
Michiel O’Herne 4387783

Course Professor: Dr. Georgios Gousios

The following report covers our minor project in big data analysis. Detailing the challenges faced and the progress made over the span of several weeks.

Table of contents:

[Project start 3](#_Toc503791621)

[Objectives: 3](#_Toc503791622)

[Motivation: 3](#_Toc503791623)

[Tools used:. 4](#_Toc503791624)

[In-depth project setup: 5](#_Toc503791625)

[Obtaining the data: 5](#_Toc503791626)

[Data sources: 5](#_Toc503791627)

[Converting obtained data: 5](#_Toc503791628)

[Analysing the data: 5](#_Toc503791629)

[Visualisation of analysis: 5](#_Toc503791630)

[Manners of visualisation: 5](#_Toc503791631)

[Used package: 6](#_Toc503791632)

[Overall opinion: 6](#_Toc503791633)

[Differences around the world: 6](#_Toc503791634)

[Differences over time: 7](#_Toc503791635)

[Popular tags: 8](#_Toc503791636)

[Results: 9](#_Toc503791637)

[Addendum: 10](#_Toc503791638)

[Similar project / methods: 10](#_Toc503791639)

[Alternate methods we could have used: 10](#_Toc503791640)

[Graphical User Interface: 11](#_Toc503791641)

[Goal: 11](#_Toc503791642)

[Used package: 11](#_Toc503791643)

[Manual: 11](#_Toc503791644)

[Installing pre-requisites: 11](#_Toc503791645)

[Launching the program: 11](#_Toc503791646)

[Using the program: 12](#_Toc503791647)

[Sources used: 13](#_Toc503791648)

# Project start

The main objective we set out to accomplish was performing sentiment analysis on the content of tweets, and to present this in several directly understandable manners.

This task can be split into three distinct sections: gathering and sorting the data, performing analysis on the data and lastly portraying this analysis in a useful medium.

## Objectives:

Our objectives for were to first be able to collect data from a twitter using a twitter spritzer. A spritzer is a kind of twitter stream that gives one percent of the tweets that meet a certain set of filters. Other types of twitter streams give more data and would be preferred but the spritzer is the only one that can be used free of charge. The only thing that is needed to be able to use this type of stream is a twitter account. The filters that can be selected include the language of the tweets, words used in tweets, hashtags in a tweet and many more. After the tweets are obtained they need to be analysed to judge whether or not the tweet is positive of negative about the subject that it’s talking about. This will be done through the usage of an open source sentiment analyser. This analyser will rate a tweets positive, negative and neutral sentiment. By combing the sentiment about a larger volume of tweets we can discover a trend in the sentiment. This trend can be anything, for instance the general sentiment about the weather is most likely better on a sunny day than on a rainy day. By plotting the general sentiment in a line graph, we can see this. Another way of visualizing these results is in a pie chart. By using this we can see whether the general sentiment on a subject was through the entirety of the time that the twitter stream was open. By using the location data that can be included in a tweet, this depends on whether or not the user has opted in to sharing his or her location data, you can also figure out what a countries sentiment is. An example of this would be that during the World Cup 2016 final, you can see that Germany would be more positive about than Argentina, especially after the 1-0 was scored. The problem with this is that not a lot of people opt in to sharing their location data. This means that this is very inaccurate and can potentially lead to no data being obtained.

## Motivation:

The idea for this project came from our interest in major sporting events. Twitter can be used to quickly find out how a group of people think about a certain subject. For instance, when a major sports event is happening, lots of tweets will be sent out with hashtags relating to that event. By filtering these tweets on hashtags and then running a sentiment analysis on the tweets, one can figure out which of the teams is the preferred team or if the game is exciting. One of the biggest sporting events every year is the Super Bowl. This is the final to the NFL season and the winner of that game is crowned champion. Around the time leading up to this event lots of tweets are sent out containing hashtags with regards to the two teams competing and a general hashtag about the Super Bowl itself. By analysing the tweets containing these hashtags we can find out which of the two teams is favoured by the fans in general and which team is favoured to win. During the Super Bowl itself we can use the sentiment analysis to determine whether or not the game is seen as exciting. If it is, the general sentiment about the game will increase. When a play ends in one of the two teams scoring points we can most likely see that in the sentiment in the tweets that contain the scoring teams hashtag. One of the two will have a very positive sentiment while the other will be very negative. The general sentiment will also be likely to be more positive in high scoring but close games, neutral fans will be more likely to enjoy this type of game, while the sentiment will be a lot less positive if the game ends up being a blowout, the fans of the winning team will be very positive about the game while neutral fans and those of the losing team will send out more negative tweets about the game.

## Tools used:.

1. **Python:** Programming language with an aptitude for data analytics tasks.
   1. **NLTK**: Natural Language Toolkit, a suite of libraries focused on the analysis of language.
      1. **Vader Sentiment Analyser:** Part of NLTK focused solely on sentiment analysis.
   2. **Plotly:** a graphing library built to provide a multitude of graphing solutions.
   3. **Tkinter:** A graphic user interface package.
2. **Twitter API:** an api supplied by twitter to allow streaming of their data.
3. **Github:** Code development platform using a web-based git version control repository.
4. **PyCharm:** an IDE tailored to the Python language

# In-depth project setup:

## Obtaining the data:

We have chosen to concurrently use two methods of data gathering, i.e. already created data sets obtained from a database, along with live streamed data directly from twitter itself. Our focus is directed more towards the streaming of tweets, as this data opens possibilities to provide real time insight into trends and public opinions. The pre-made datasets however form a way for us to test our analysing tools in a consistent method.

### Data sources:

Twitter has set up its own api that allows for live streaming of tweets in real-time, as such our data stream is built around this api. The output of Twitter their api is encoded in JSON, containing not only the tweets themselves, but also all possible related data such as number of like, retweets and reactions tweets have received.

### Converting obtained data:

As Twitter supplies their data in a JSON format, and our sentiment analyser of choice operates on a per line basis, the conversion needed to go from input data to analysable data is rather efficient. In the file “sort\_for\_graphing.py”, the function “sort\_into\_list” pulls a tweet out of the JSON file containing all data obtained from twitter, and writes this into a list, alongside the timestamp of the tweet and the country from within which the tweet was sent.

## Analysing the data:

the tweet was sent.

* Divide the data into positive - negative tweets
* Give the overall opinion of the public (i.e. twitter users) of the topic
* Give the popularity of given subject

Tools used to sentiment analyse the data  
used environment / packages

One of the first things that becomes apparent is the popularity of a subject. This determined by simply counting the number of tweets that contain a hashtag about that subject. After that all the tweets are analysed for their sentiment and this will give information about both the sentiment of the induvial tweet and the general sentiment.

## Visualisation of analysis:

### Manners of visualisation:

When the data is collected, the only step remaining is the visualisation of it. Therefore it is necessary to first determine what to achieve. A few things must be made visible, and the goal is to determine the best way to do that.

1. See the overall opinion about a specific topic
2. See if there are differences in the opinion about a topic around different countries in the world
3. See whether the opinion on a topic changes over time

For the first goal, the best way to display the overall opinion would be a pie chart divided into three part; a negative, neutral and positive part. To display possible differences around the world, the best way to display this is obviously on a world map. The third and last goal is to display is the opinion changes over time. This can be done in several ways, for not that much messages (less than twenty) the best way is probably by displaying the data as a bar chart, where every bar represents a message and consist of a negative, neutral an positive part. When all bars are plotted on the x-axis (which is time), it is possible to see if which parts (positive, neutral, negative) are increasing and decreasing. However, when there is a lot of data available about a specific topic (over a long period) the bars can become very tiny and it is not easy anymore to see if the opinion really changes over time. If that is the case, it is better to create a scatterplot of the messages, with a moving average plotted as a line.

### Used package:

To visualize the data, a package called “plot.ly” is used. This is an open source tool to create easy and clear different interactive graphs in Python, which are displayed in the web browser. When using the offline version, a resulting plot is stored on the local disk as a html-file, which can be opened by almost any device. For generating the plot most of the time there is no internet connection required, when displaying a world map however, the map is downloaded from <http://cdn.plot.ly/>.

### Overall opinion:

When using a pie chart it is immediately clear what the overall opinion about a specific topic is. This is done by summing up all the negative, neutral and positive parts of the messages. A pie chart is shown in Figure 1 below.

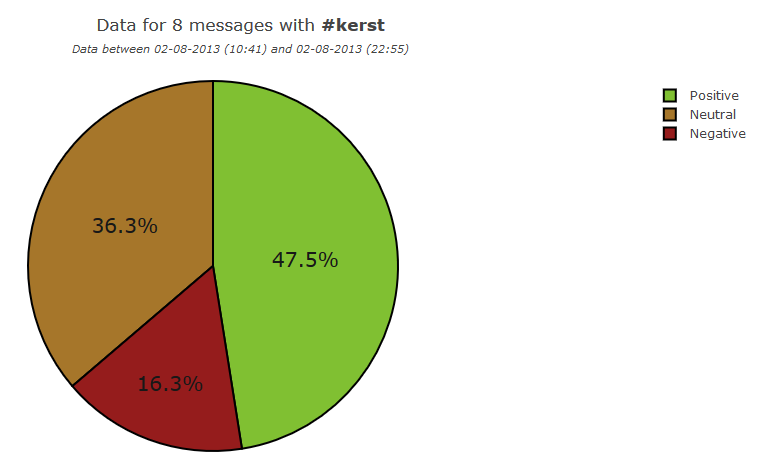


Figure 1: Pie chart shows the overall opinion

It can sometimes also be better, to not take every part (negative, neutral, positive) into account, but instead classify a message in one of the three categories (or just negative or positive by ignoring the neutral class). This mode is called “hot one” and the mode of the pie chart can be passed through by use of an extra argument. This functionality is implemented in the file *“plot\_pie.py”*.

### Differences around the world:

To see if there are different opinions in different countries around the world about a specific topic, a world map is used to plot that. For each country an average is taken of the compound and the number of messages from each country is counted and stored. Which is finally plotted on the world map and should like the figure depicted in Figure 2 below. The darker the colour of a country, the more messages were received from this country when displaying the occurrence. When displaying the compound, red means a negative opinion and green a positive one. This functionality is implemented in the file *“plot\_world.py”*.

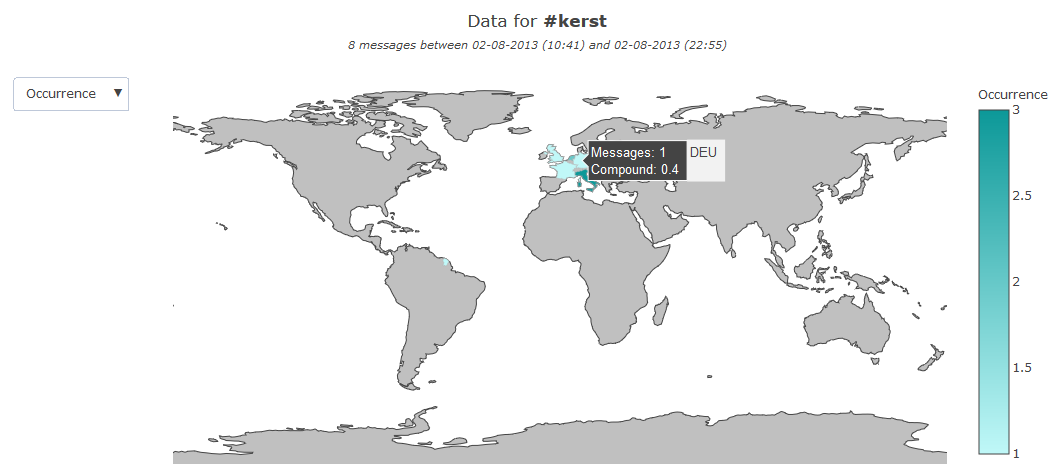


Figure 2: World map shows differences around the world

### Differences over time:

To determine how the opinion about a specific topic is changing, a bar chart can be a nice way to display this possible change. With on the x-axis the time and on the y-axis the interpretation of the message in negative, neutral and positive parts. When more than one message about the given topic is received, an average is taken. This gives (for not too many messages) a nice overview. This functionality is implemented in the file *“plot\_bar.py”*.

However, when there are a lot of messages received, reading from the bar chart can be quite complicated and this is where scatter plot comes in. Again is the time drawn on the x-axis and the opinion on the y-axis (the higher the compound, the more positive a message is). Every dot represents a message and a dotted line is plotted with the average. To see the changing of opinion even better, a moving average is also drawn as a green line. This moving average takes the average opinion of all the massages within a minute/ hour. This setting can be set by passing an argument to the function. An example of this plot is shown in Figure 3. This functionality is implemented in the file *“plot\_line.py”*.

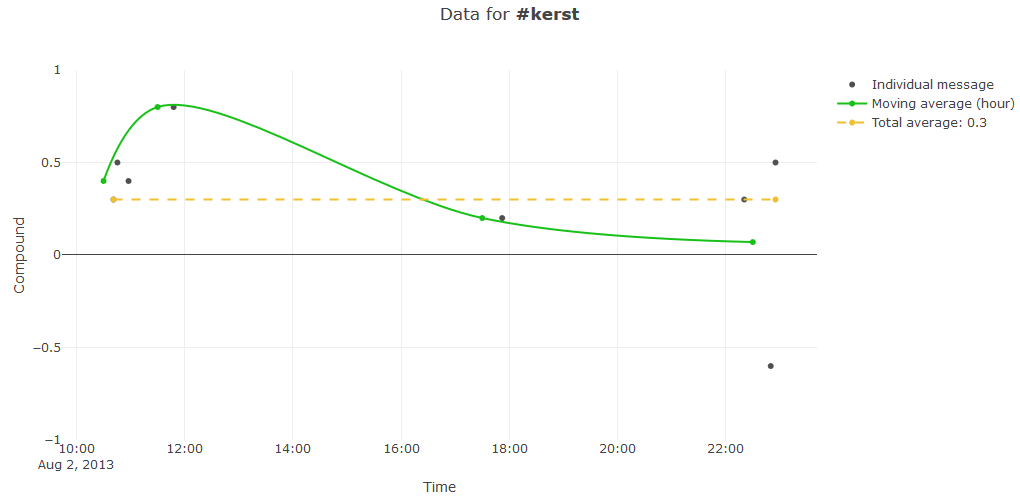


Figure 3: Line/ scatter plot shown the different opinion over time

### Popular tags:

Before setting a hashtag on what topic to display, it is possible to plot a bar chart with the sixteen most popular tags in the collected data, sorted in descending order. An example of this plot is shown in Figure 4. This functionality is implemented in the file *“plot\_overview.py”*.

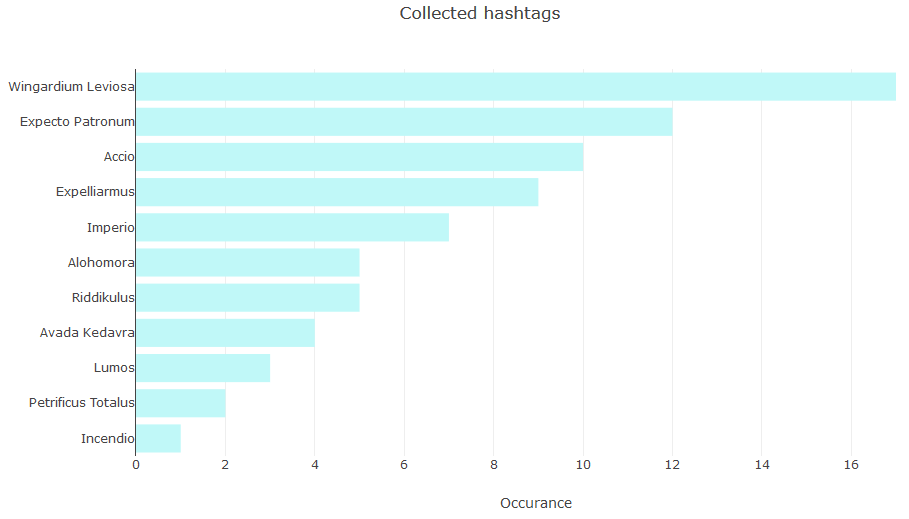


Figure 4: Bar chart with the most popular hashtags

# Results:

Couple of graphs + part of tweet data set?

# Addendum:

## Similar project / methods:

Multiple project setup to do twitter analysis

Mainly business focused models renting their services for research and development purposes

## Alternate methods we could have used:

Spark streaming

# Graphical User Interface:

## Goal:

The goal of the graphical user interface (GUI) is to make the use of the program more user friendly than writing on the command line. The GUI will consist of three different sections; first the part of collecting the data, then a part with options how to display the data, and finally steps to actually display the data.

## Used package:

To make a clear GUI in Python, a package called “Tkinter” is used. With this package it is possible to make buttons that can trigger different functions, as well as fields where the user can enter their own input.

## Manual:

Here follows a manual on how to use the program.

### Installing pre-requisites:

To start using the program a few things need to be installed. To start, the following python modules: “Plotly” and “tweepy”. Next the sentiment analyser package “Vader sentiment analyser” from the python module “NLTK” should be installed. This is achieved by running the included file named “install\_vader.py” and selecting the option “Download” on the interface that pops up.

### Launching the program:

To launch the program, run the file named “gui.py”. This will open up the graphical user interface (GUI) of the program. This window should look like Figure 1 below.

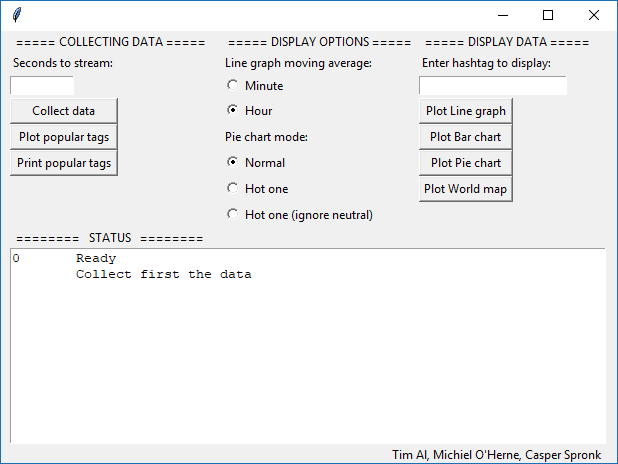


Figure 1: The GUI of the program

### Using the program:

The window contains four different sections. One for “Collecting data” on the top left, the “Display options” in the top middle, the “Display data” on the top right and at the bottom the status is displayed.

Start by collecting data from Twitter. This can be done by setting a number of seconds to the field at the top left. If this value is set, press the button “Collect data” below the box and the program will gather some data. Please note that the process of collecting data cannot be interrupted and Windows can state that while this process is running the program is not responding. When the data in the time interval is collected, the total number of received messages is shown in the status box.

To see which tags are collected, it is either possible to print the five most popular tags in the status box by pressing the “Print popular tags” button, or the sixteen most popular tags are displayed in a horizontal bar graph. In both cases, the number of messages per tag is also displayed.

To display the data, you first have to enter a so called “hashtag” in the field on the top right, the data of the given hashtag will be plotted when pressing one of the four buttons to plot the data. Note that the hashtag-field is case-sensitive. The data can be displayed in four different ways:

* **Line graph**: every message is represented by a dot in a scatter plot. The x-axis is the time and the y-axis the compound. The closer to one, the more positive the message was. The average compound of all the messages is drawn as a yellow dotted line. The green line is a moving average, either per hour (default) or per minute. This setting can be set *before* plotting the data under “Display options”.
* **Bar chart**: all messages (or the average if multiple messages are received in the same second) are displayed as a bar on the x-axis. Each bar can contain a red (negative), yellow (neutral) and green (positive) part, which indicates how positive the message is rated.
* **Pie chart**: an overall opinion of the given topic (hashtag) is displayed. This can be done in different ways. The normal mode, which is the average of each negative, neutral and positive rating in a message. In the hot-one mode, only the highest value of the negative, (neutral) and positive is selected. The mode can be set under “Display options” *before* plotting the data.
* **World map**: The data is drawn in a world map. In the world map, first select on the top left which data should be displayed in the map. This can be the occurrence (how many messages with the specified topic are send from each country) or the opinion (red country means a negative opinion, a green one a positive). Please note that only very few messages received from Twitter contain a location.

In every plot there are a lot of options available to analyse the data. Hovering on data can display more information, and with the controls at the top right it is possible to zoom on specific parts of the data. There is also an option to export the data in different formats.

# Sources used:

setup of project:  
<Https://marcobonzanini.com/2015/03/02/mining-twitter-data-with-python-part-1/>

examples how to sort data:   
<https://www.analyticsvidhya.com/blog/2014/07/world-cheering-2014-fifa-wc-winner-twitter/>

python based twitter apis:   
<https://developer.twitter.com/en/docs/developer-utilities/twitter-libraries#python>

twitter sentiment analysis example using spark:   
<https://mapr.com/blog/spark-streaming-and-twitter-sentiment-analysis/>

tweet import using spark:   
<https://medium.com/@harinilabs/day-6-realtime-tweets-analysis-using-spark-streaming-with-scala-10a9937aae57> <https://github.com/harinij/100DaysOfCode/tree/master/Day%206%20-%20Spark%20Streaming%20using%20Scala>

realtime tweet analysis using spark:   
<https://www.toptal.com/apache/apache-spark-streaming-twitter>

spark java guide importing tweets:   
<http://ampcamp.berkeley.edu/big-data-mini-course/realtime-processing-with-spark-streaming.html>

big data for noobs:   
<https://rawkintrevo.org/2016/09/30/big-data-for-n00bs-my-first-streaming-flink-program-twitter/>

flink how-to diagrams:   
<https://github.com/wdm0006/flink-python-examples>

Old Twitter data to test algorithms:   
<https://archive.org/details/archiveteam-json-twitterstream> <https://archive.org/search.php?query=collection%3Atwitterstream&sort=-publicdate>

twitter "spritzer" definition:   
<http://www.adweek.com/digital/twittery-things-you-forgot-existed/#/>

NLTK in Nederlands:  
<https://stackoverflow.com/questions/40212895/nltk-tag-dutch-sentence>

Introduction to NLTK (21 parts): <https://www.youtube.com/watch?v=FLZvOKSCkxY&list=PLQVvvaa0QuDf2JswnfiGkliBInZnIC4HL>

most used English words:   
<http://techland.time.com/2009/06/08/the-500-most-frequently-used-words-on-twitter/>

small bit on geo streaming:   
<https://cmry.github.io/notes/twitter-python>

matlibplot:   
<https://stackoverflow.com/questions/11874767/real-time-plotting-in-while-loop-with-matplotlib>

Visualizing data with plot.ly:   
<https://plot.ly/python/reference/>

500 popular words:  
<http://techland.time.com/2009/06/08/the-500-most-frequently-used-words-on-twitter/>