#Emotionalyze

Final Report
&
User Manual

Mikey Farren **Chris Giuffrida**

Thomas Krill **Pedro Saunero**

4 May 2017

Welcome to #Emotionalyze:

#Emotionalyze is a web application designed to give users insight into how the country is feeling about a given topic. Leveraging the data from Twitter and the power of IBM's Watson API, we are able to provide quantitative data on a world of emotion!

Retrieving #Emotionalyze Source Code:

All of the code needed to run #Emotionalyze is stored on GitHub. The following link should bring you to the repository. From there, you can download the code. You will need all of the files in the repository for #Emotionalyze to run correctly.

https://github.com/ChrisGiuffrida/DataStructuresProject

Getting Started with #Emotionalyze:

#Emotionalyze requires the user to install several python libraries in order to make the code run correctly.

Watson Developer Cloud Python SDK:

To install the **Watson Developer Cloud Python SDK**, simply run the following command in your terminal (you might need to run the command using "sudo):

```
$ pip install --upgrade watson-developer-cloud
```

To see more information about **Watson Developer Cloud Python SDK**, please go to the follow link:

https://github.com/watson-developer-cloud/python-sdk

Tweepy:

To install the **Tweepy** python library on your computer, simply run the following command (you might need to run the command using "sudo):

```
$ pip install tweepy
```

To see more information about **Tweepy**, please go to the following link:

https://github.com/tweepy/tweepy

Tornado Web Server:

To install **Tornado Web Server**, simply run the following command (you might need to run the command using "sudo):

```
$ pip install tornado
```

To see more information about **Tornado Web Server**, please go to one of the following links:

http://www.tornadoweb.org/en/stable/index.html

https://github.com/tornadoweb/tornado

Colour:

To install the **Colour** python package, simply run the following command (you might need to run the command using "sudo):

```
$ pip install colour
```

To see more information about **Colour** please go to one of the following links:

https://pypi.python.org/pypi/colour/

Running #Emotionalyze

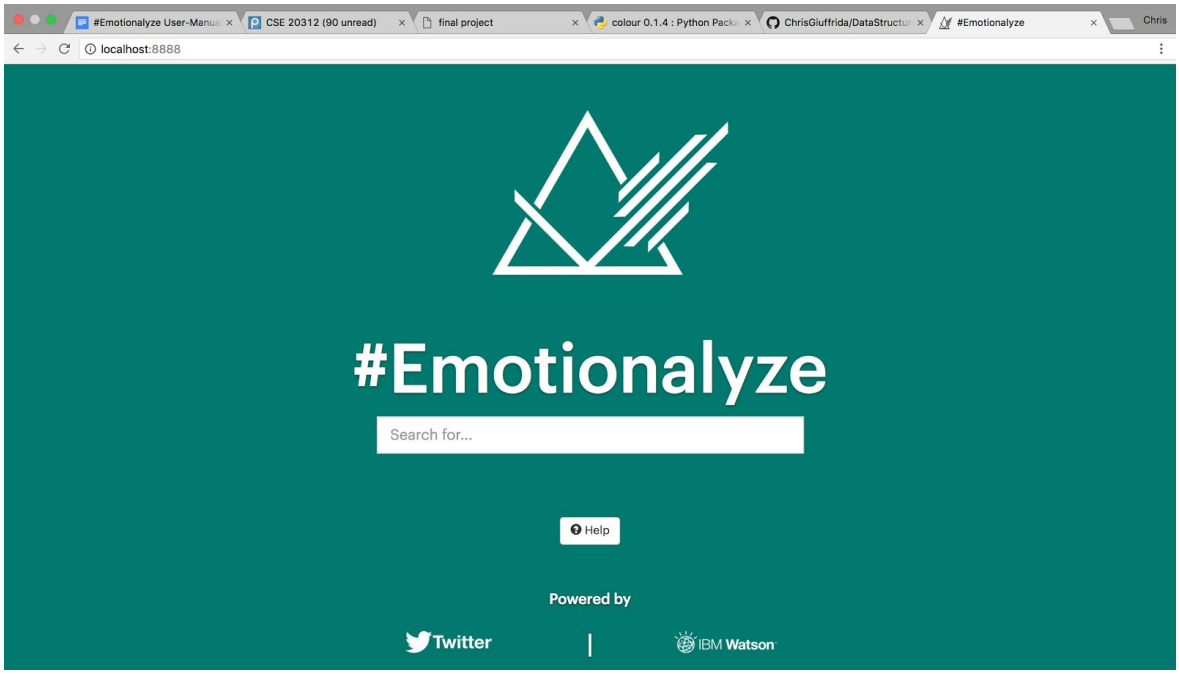
To run #Emotionalyze, please follow the steps listed below:

1. Open a terminal on your computer and navigate to the project folder you downloaded from the project's GitHub repository (linked above).
2. Inside the project folder, you should see two folders, namely "`PythonScripts`" and "`Website`".
3. Move into the "`PythonScripts`" folder.
4. Check to see if you have the correct permissions for each of the scripts. You will likely have to run "`chmod 755 [name of script]`"for each of the three Python scripts in the folder.
5. Run the "`EmotionalyzeServer.py`" script (`./EmotionalyzeServer.py`). You must let this run in the background while you use the web application.
6. Open a web browser, preferably Google Chrome.
7. Type `localhost:8888` into the browser.
8. You should see the #Emotionalyze home page appear. You are ready to use #Emotionalyze!

Using #Emotionalyze

Homepage:

Once you see the #Emotionalyze home page, you are ready to begin using the application! On the home page, you will see a search bar and a help button, as shown in the picture below. A description of how each element works follows the picture.



Help Button:

Clicking on the help button will trigger a popup window. Inside the popup window is a brief description of what #Emotionalyze is and how to use it. You can click either the “×” or the “Close” button to exit the window and return to the application.

Search Bar:

The search bar is where you are going to enter a search term for #Emotionalyze to analyze. The search term MUST be five words or less. Searches greater than five words will not be accepted, and the main page will reload with your original search term deleted. Once a valid search term is provided, press the “Enter” button on your keyboard to load the results page.

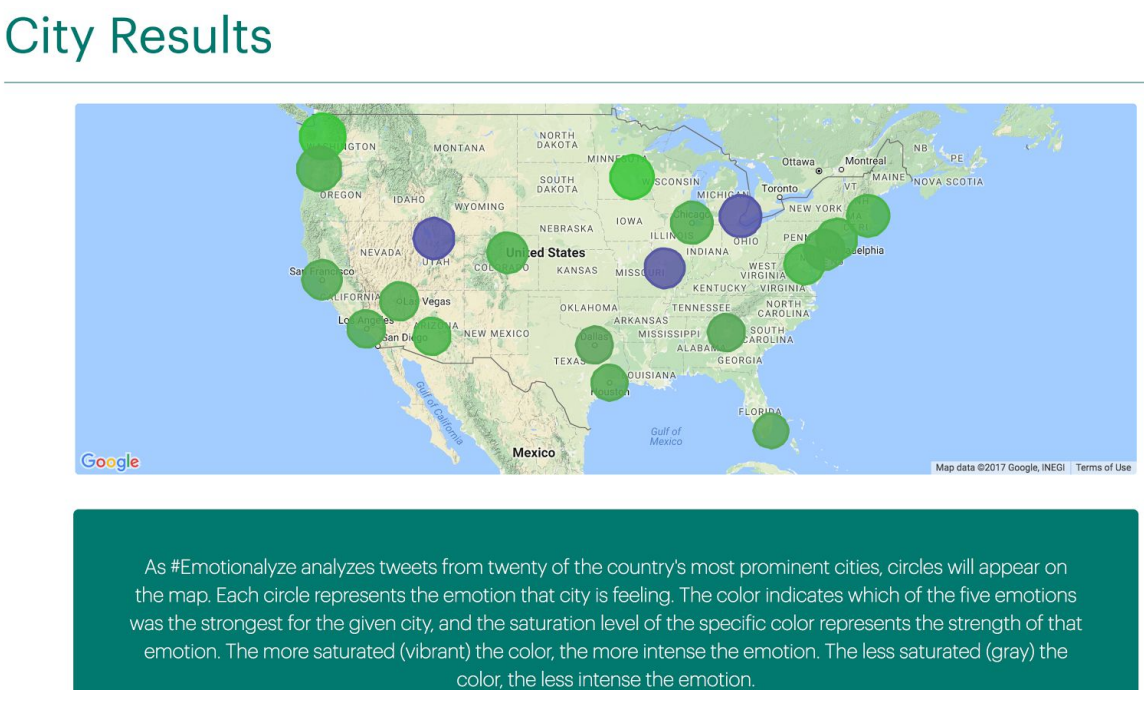
Results Page:

With a valid search term entered, the results page should load. On the results page you will see a navigation bar, a section for city results, and a section for real-time results, as shown in the pictures below. A description of how each element works follows each picture.

Navigation Bar

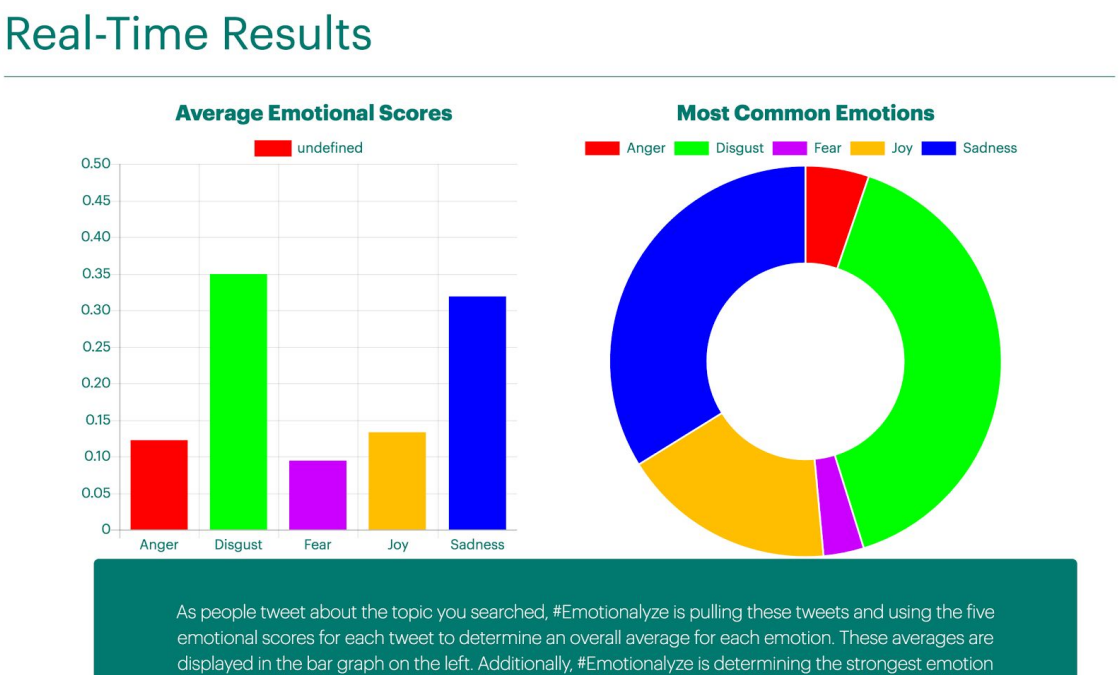
The Navigation Bar has three links. The first link is the #Emotionalyze application name. This link was designed to bring the user back to the home page to enter a new search. However, with the current design of the project, this link does not fully work and new searches can only be conducted by restarting the server. The second link is “About.” Clicking on “About” will enable a pop-up menu for the user to read about the program and better understand the results they are seeing. Lastly, there is the “help” link. The “help” link opens a different pop-up menu that explains why results may not be showing up. Both pop-up menus have close buttons to close out of the window.

City Results



The city result section of the results page uses a Google Map. When the page first loads, circles will start to appear on the map. Each circle encompasses a major city, with the circles loading in the order of population of the cities. The color of the circle represents the most prominent emotion that city is feeling towards the search topic, using recent tweets as the data source. Further, the saturation level of the color filling the circle reflects the level of intensity for that emotion. The higher the saturation level (vibrant), the more intense the emotion. The lower the saturation level (gray), the less intense the emotion. When all twenty circles have appeared, the map will stop outputting new information, but the map will retain its state until a new search.

Real-Time Results



At the same time the map is populating, #Emotionalyze is pulling tweets in real-time from an open Twitter stream. The tweets are then analyzed. Each tweet comes back with an emotional score from 0.0 to 1.0 for each of the five emotions, anger, disgust, fear, joy, and sadness. Each tweet's score for each emotion is used to determine an overall average score for each emotion. These average emotional scores are displayed in the bar graph on the left. Additionally, #Emotionalyze finds the strongest emotion for each tweet and keeps track of a count for the number of tweets that have each emotion present as the strongest. These total counts are displayed in the pie chart on the right.

Known Bugs

There is only one major bug in the overall application, namely the ability to enter a new search without restarting the server. In order to leave the results page and enter a new search, you must follow the steps below:

1. Close the browser tab that has #Emotionalyze open on `localhost:8888`.
2. Go to the terminal and suspend the EmotionalyzeServer.py script by typing `Ctrl+Z`.
3. Kill the process with the kill command and the process ID of the script.
4. Restart the `EmotionalyzeServer.py` script.
5. Reopen `localhost:8888` and enter your new search term.

There are no other major bugs we are aware of. If you try searching an irrelevant term, the program will simply draw a gray circle around each of the cities, indicating that there was not enough tweets available to properly analyze the emotion. If some cities return tweets but others do not, #Emotionalyze will correctly analyze only the cities it has enough data for, drawing gray circles around the cities with too few tweets. Additionally,, if you try to search something greater than five words, #Emotionalyze will return to the main page and blank your search, expecting you to enter a smaller search phrase. Lastly, the program will never crash. As long as the server is running and you do not hit the back button on your browser, the application will continue to update the graphs in real-time for as long as you stay on the page.

Additional Information

The GitHub repository for the project also includes a “testEmotionalyze.py” script that runs a simple test program to check the validity of the main functions we wrote and use in the project.

Further Questions

If the program is not running correctly on your computer and or if you have any general questions about the project, please send an email to:

cgiuffri@nd.edu

Group Meetings

Date	**Time**	**Task**
21 March 2017	4 h	Figuring Out APIs
26 March 2017	3 h	Figuring Out APIs
2 April 2017	3 h	Completing Python Scripts
3 April 2017	2 h	Completing Python Scripts
30 April 2017	8 h	Updating Map and Charts in Real Time
1 May 2017	2 h	Creating Presentation
2 May 2017	3 h	Fixing Bugs
3 May 2017	0 h 45	Meeting with Prof. Kumar
4 May 2017	4 h	Lab Reports, Finishing Touches
Total:	**29 h 45**	

Individual Timelogs

Mikey Farren

Date	Time	Task
19 March 2017	4 h	Learning Python the Hard Way
4 April 2017	2 h	Learning HTML and JavaScript
9 April 2017	2 h	Code Academy HTML Tutorial
N/A	*29 h 45*	*Group Meetings*
Total:	**37 h 45**	

Chris Giuffrida

Date	Time	Task
13 March 2017	2 h 30	Learning Python the Hard Way
15 March 2017	4 h 30	Learning Python the Hard Way
16 March 2017	1 h	Udacity Intro to HTML
17 March 2017	4 h	Exploring APIs, writing initial code
7 April 2017	5 h	Learn HTML and Bootstrap
10 April 2017	5 h	Develop Website
12 April 2017	4 h	Develop Website
24 April 2017	5 h	Re-formatting Website
3 May 2017	4 h 30	Fixing website bugs
4 May 2017	2 h	Wrote user manual and added pop-up windows for “Help” and “About” links.
N/A	*29 h 45*	*Group Meetings*
Total:	**68 h 15**	

Thomas Krill

Date	**Time**	**Task**
14 March 2017	3 h	Learning Python the Hard Way
15 March 2017	2 h	Learning Python the Hard Way
16 March 2017	2 h	Learning Python the Hard Way
17 March 2017	4 h	Exploring APIs, writing initial code
8 April 2017	4 h	Learning HTML and Bootstrap
10 April 2017	5 h	Develop Website
24 April 2017	5 h	Re-formatting Website
3 May 2017	1 h	Overriding Bootstrap Default Font
N/A	*29 h 45*	*Group Meetings*
Total:	**55 h 45**	

Pedro Saunero

Date	**Time**	**Task**
18 March 2017	2 h	Reading Twitter Stream API Documentation
19 March 2017	2 h	Reading Twitter REST API Documentation
7 April 2017	4 h	Researching Tornado Web & HTML
16 April 2017	3 h 30	Configuring Tornado Web Server
8 April 2017	1 h 30	Connecting Python and Google Maps API
27 April 2017	3 h	Fixing Static file Handler to serve CSS/images
28 April 2017	5 h	Reconfiguring Tornado Server and program structure/Added Handlers
N/A	*29 h 45*	*Group Meetings*
Total:	**50 h 45**	