

Design Document
Sentiments Analysis with Twitter
Team Jazz Men

Anagh Goswami 1217426

Meet Pandya 1214306

Jasman Gill 1211554

Jesse Truong 1222722

Jia Xu 1213268

January 11, 2017

Contents

1	Revision History	1
2	Introduction	2
2.1	Purpose	2
2.2	Description	2
2.3	Scope	2
3	Overview	3
4	Development Details	3
4.1	Language of Implementation	3
4.2	Supporting Frameworks/APIs/Library	3
5	Implementation Components	4
5.1	Flask Function	4
5.2	Get Data Function	4
5.3	Twitter Search Function	4
5.3.a	Alchemy API	5
6	Error Handling	5
6.1	Server Failure	5
7	User Interface Elements	6
7.1	Home Page	6
7.2	How It Works	6
7.3	Demo	7
7.4	Results	8
8	System Components	9
8.1	Comparative Analysis	9
8.1.a	Feature Detail	9
8.1.b	Feature Implementation	9
8.2	Graphical Data Representation	10
8.2.a	Feature Detail	10
8.2.b	Feature Implementation	10
8.3	Historical Data - BackEnd	11
8.3.a	Feature Detail	11
8.4	Track User Details of Specific Tweets	11
8.4.a	Feature Detail	11
9	JSON Data Format	12
10	Algorithms	13

11 Solution to Challenges	13
11.1 Challenges	13
11.2 Solution	13

1 Revision History

Table 1: Revision History

Description of Changes	Author	Date
Created first version with appropriate sections.	Meet Pandya	2017-01-07
Edited the document	Jesse Truong	2017-01-08

2 Introduction

2.1 Purpose

The purpose of Sentiments Analysis with Twitter is to provide a way for high school students to see what is being said about a university on the social media. This project also provides benchmark analysis which would help student compare and contrast two potential universities of their choices.

2.2 Description

Twitter is a hugely popular social media application used to communicate people's thoughts and opinions with those around them. There are currently a whole generation of students that use the application to see what others think of a certain topic. Some of these students being highschool seniors who are trying to decide what university they are going to attend. There is no medium for them to be able to quickly assess their fellow peers thoughts on certain schools other than doing independent research. Allowing students to see collected and analyzed data would greatly help them in being able to determine which university is right for them. This is even more so true for students who are on the edge of making a decision, but would like to see others opinions be it negative or positive to help them with their own. Depending upon how successful the project is, it could possibly even be used by companies to determine how they can improve and what others around them are doing that is seen as positive features.

2.3 Scope

The scope of Sentiments Analysis is started with but not limited to High school students. This has a potential to be expanded to organizations, brands and many more categories where Sentiments Analysis could be utilized for the intended purpose.

3 Overview

4 Development Details

4.1 Language of Implementation

The source code of the Sentiments Analysis is written with Python using the Flask Web Framework. The Webpage UI used to display the results is developed using HTML, CSS, Bootstrap Framework and Javascript.

4.2 Supporting Frameworks/APIs/Library

Alchemy Language API

Twitter API

Requests Library for HTTP (Python)

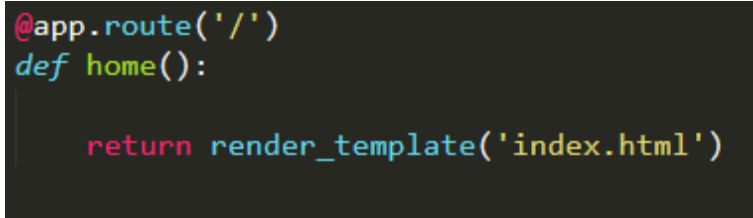
Flask Web Framework

Bootstrap Framework

5 Implementation Components

5.1 Flask Function

Flask Web Framework is used to build the webpage using python. The "home" function within the code for Flask calls upon the HTML file that encapsulates all the Front End code for the homepage of the UI. The next function called "universities" calls upon "getdata" function to retrieve the Sentiments scores and returns the data in JSON format as seen in Figure 1.



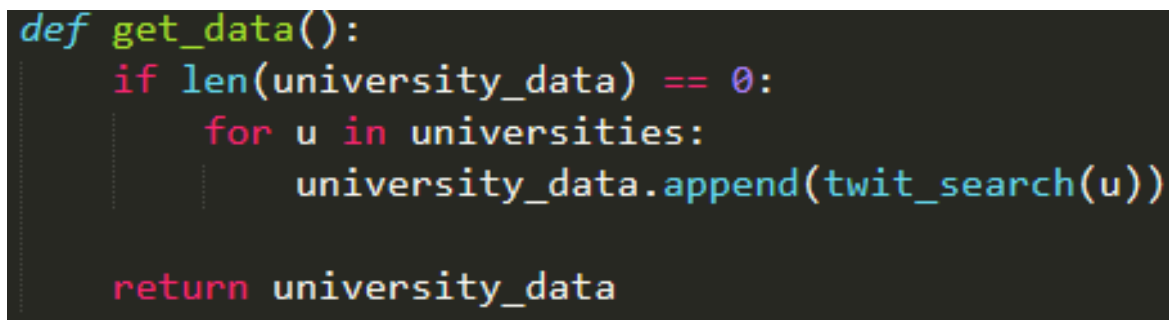
```
@app.route('/')
def home():

    return render_template('index.html')
```

Figure 1: Flask Function

5.2 Get Data Function

The "getdata" function as defined, iterates through the query "keyword" and each letter is fed as the input to the Twitter Search Function described in the succeeding section. Each letter is inputted in the Twitter Search function and the resulted data is added to a "storage variable". By the end of the loop, the "storage variable" has all the data retrieved from the Twitter Search Function. Finally, the "storage variable" is what is returned in this Get Data Function. The functionality of Get Data can be seen in Figure 2.



```
def get_data():
    if len(university_data) == 0:
        for u in universities:
            university_data.append(twit_search(u))

    return university_data
```

Figure 2: Screen image of Get Data Function

5.3 Twitter Search Function

In this section the twitter search function will be discussed that entails using the Twitter API and also the Alchemy API. The twitter search function has an input

parameter which is the query 'keyword' to be searched for on twitter. This keyword is used as the query in the URL for the Twitter API. Requests library is used to get the tweets from twitter API and the resulted data is stored in a variable using JSON format as seen in Figure 3.

```
def twit_search(query):
    url = 'https://api.twitter.com/1.1/search/tweets.json?q='+query
    oauth = get_oauth()
    json_obj = requests.get(url, params={'count':100, 'lang': 'en'}, auth=oauth)
    data = json_obj.json()

    tweet_list = {'positive': [], 'negative': [], 'neutralCount': 0, 'score': 0}
```

Figure 3: Screen image of Twitter Search Function

5.3.a Alchemy API

Within the Twitter Search Function, the Alchemy API is called upon to feed the tweets in order to retrieve the Sentiment score. The score is tallied based on its emotion whether it is neutral, positive or negative. Neutral scoring tweets are not accounted for in the final result as they do not make a difference. The snippet of code for the Alchemy API is provided in Figure 4.

```
# print item['user']['screen_name'], item['id_str']
sentiment = alchemy_language.sentiment(text=item['text'])
if sentiment['status'] == "OK":
    analysis['sentiment'] = sentiment['docSentiment']['type']
    if analysis['sentiment'] == "neutral":
        tweet_list['neutralCount'] += 1
    else:
        analysis['score'] = sentiment['docSentiment']['score']
        tweet_list[analysis['sentiment']].append(analysis)
        tweet_list['score'] += float(sentiment['docSentiment']['score'])
```

Figure 4: Screen image of Alchemy API function

6 Error Handling

6.1 Server Failure

- Backup and Recovery Scripts
- Condition to redirect traffic when server is down

7 User Interface Elements

In this section the different elements of the Web UI are displayed with images and brief description.

7.1 Home Page

This is the top of the Home Page where the user begins from every time using the application.

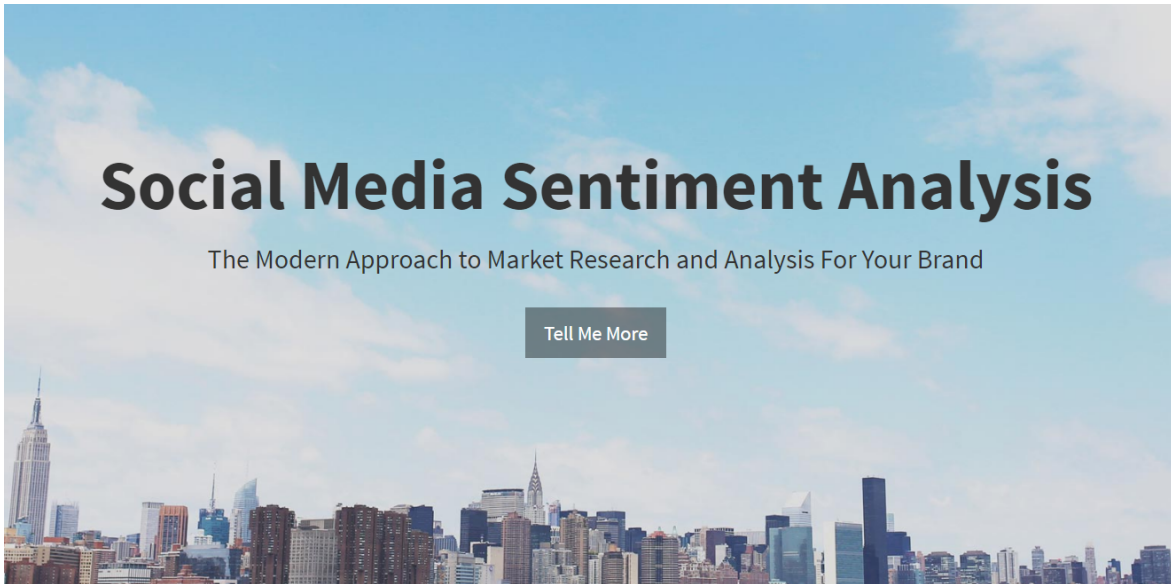


Figure 5: Top of Home Page

7.2 How It Works

After scrolling down on the home page, this section gives a brief description of how the Sentiments Analysis works

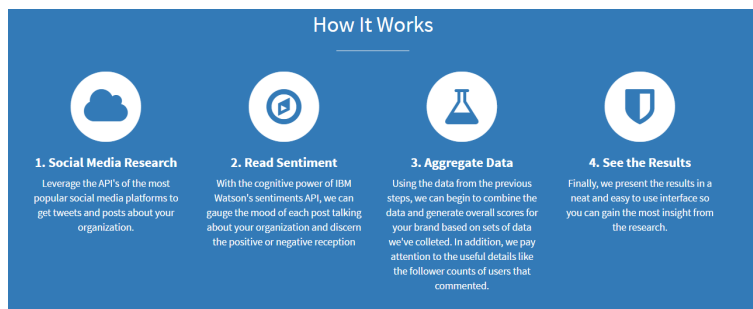


Figure 6: How it works

7.3 Demo

After further scrolling down on the home page, the Demo section shows four images and by clicking on the top right image. A demo result of data is shown further down on the home page.

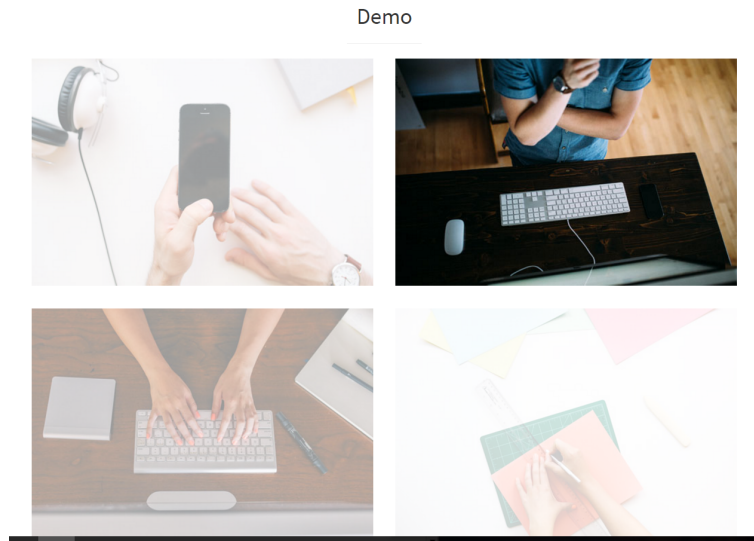


Figure 7: Demo

7.4 Results

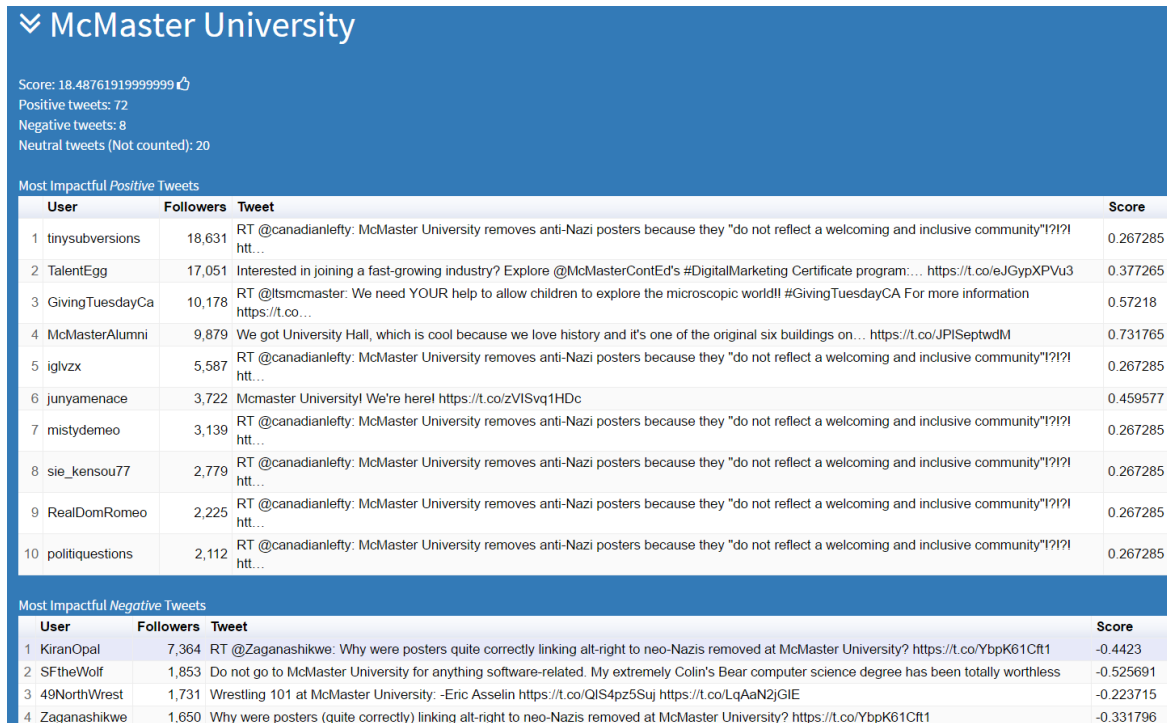


Figure 8: Demo

8 System Components

A description of the System Components of Sentiments Analysis is provided here in detail.

8.1 Comparative Analysis

8.1.a Feature Detail

Comparative Analysis is a feature that will be implemented in this project. This analysis can be applied to this project which allows users to compare and contrast Sentiments Score from two different data sets. For example, the two different data sets could be comparing two universities, where a student wants to see how much better is one from another. This allows for flexibility with how a user is able to utilize the data made available from the Sentiments Analysis. Instead of having simply one way of showing results with one query, comparative analysis gives users more ways of looking at the data.

8.1.b Feature Implementation

The comparative results will be displayed in two categories side by side, based on the query string. The result will show the query string in huge text and underneath would be the statistical numbers displayed such as the Sentiments Score, Positive Tweets, Negative Tweets and Neutral Tweets in numerical format initially. Figure 5 shows how the comparative data will be displayed on the Web UI.



Figure 9: Comparative Search Result

Next, by clicking on the Query String user will see an expanded table with a list of most impactful positive and negative tweets for that particular query string. This will allow the user to see in detail what the tweets actually discussed about the topic searched. The table will categorize each impactful tweet by user, followers, tweet and the score. Figure 6 shows how the tabular format is displayed.

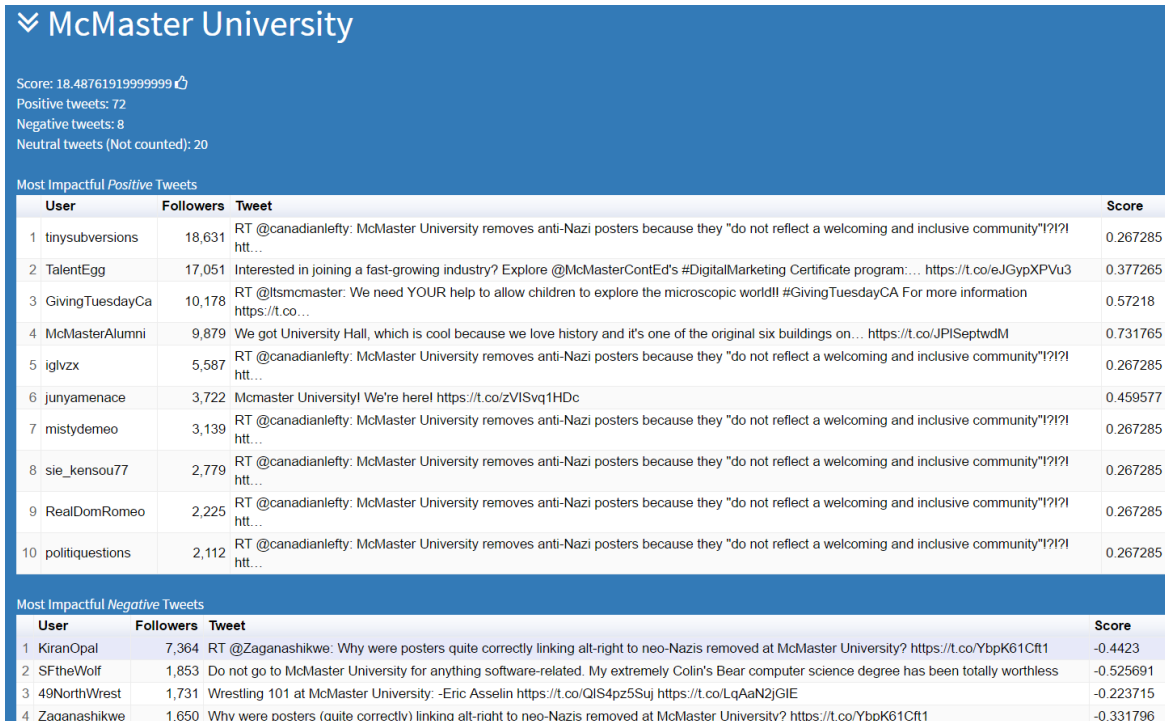


Figure 10: Comparative Search Result

8.2 Graphical Data Representation

8.2.a Feature Detail

Graphical representation of the data collected from the users interest keywords will be displayed on the webpage. Using Google Charts API, this feature will provide easy-to-read and useful graphs of the data collected. It can be used to enhance the Comparative Analysis feature by providing graphs that show the two data sets. (If we find a way to store historical data, this feature can be implemented —_i) Using historical sentiment scores, this feature can provide charts that show the sentiments trend of the keywords searched over a period of time as well. Tracked sentiments will be displayed as an overall score for each keyword searched, creating a ranking amongst the keyword set.

8.2.b Feature Implementation

The visual implementation of this feature will be used within a few different functions of the webpage. As mentioned above, the Comparative Analysis feature will greatly benefit from graphical representations of the Twitter data found from the keywords searched. Pictographs of the percentage of positive/negative sentiments towards a certain keyword can depict the overall score in a clear fashion.

8.3 Historical Data - BackEnd

8.3.a Feature Detail

Information will be stored in a database (Mongo DB) to be used for future historical data comparison. The data that will be stored are the keywords that were used as well as associated sentiment scores for tweets. The collected data can later be pulled and analyzed with respect to future keyword search requests as well as compared through different time spans to show changes in the sentiment scores to develop trends.

8.4 Track User Details of Specific Tweets

8.4.a Feature Detail

Users will input a keyword to search through Twitter messages. These messages will then be given a score depending on the sentiment towards the topic. The collected tweets will be sorted in ascending or descending order to view the extremities at the top as well as given an overall view of how positive or negative the public's opinion is on the topic. These tweets will then be taken into account for the amount of followers the user has (their overall social media influence) and will be stored to be used for other functionalities of the web application such as advanced analysis of certain topics. We will also allow directly contacting the handlers of the tweets in order to delve further into detail of why their messages have such extreme sentiment scores.

9 JSON Data Format

The data set of searched tweets are collected in JSON format. Currently, the data set is updated according to the interest of the user Below is an excerpt showing the JSON format from the data collected through the Twitter API:

```
[
  {
    "negative": [
      {
        "followers": 11678,
        "score": "-0.579868",
        "sentiment": "negative",
        "tweet": "RT @yung_mung: @bendykoval pretty sickening how forgotten Suharto's legacy has become by the Western left. Always on about Pinochet but ne\u0026",
        "user": "bendykoval"
      },
      {
        "followers": 11624,
        "score": "-0.619831",
        "sentiment": "negative",
        "tweet": "RT @Aristokles11235: Pardon me for standing up for western civ against the impending idocracy, far left OR faux right.\nOn second thought do\u0026",
        "user": "Costofles"
      },
      {
        "followers": 9458,
        "score": "-0.743297",
        "sentiment": "negative",
        "tweet": "RT @chrisderrick1: The Tainted Dollar \nTellin' a story the way the Old West used to be\nhttps://t.co/3NIfJmVnZV\n#JakeBase #Texas #western\nR\u0026",
        "user": "JSYorkshire"
      },
      {
        "followers": 3291,
        "score": "-0.760624",
        "sentiment": "negative",
        "tweet": "RT @SoniaKatiMota: Republicans call UniversalHealthcare Communism except they fail 2 add all Western Civilizations have, except f'ckin' Ame\u0026",
        "user": "QlaraQontra"
      },
      {
        "followers": 2063,
        "score": "-0.366874",
        "sentiment": "negative",
        "tweet": "RT @singofhisgrace: @LiteraryLucifer @RT_com Yes, it does. Our western news isn't accurate when compared worldwide. Not by MILES.",
        "user": "LiteraryLucifer"
      },
      {
        "followers": 1924,
        "score": "-0.328925",
        "sentiment": "negative",
        "tweet": "You can't be a Nationalist in most current-day Western countries and a Catholic. It's simply not possible.",
        "user": "GreyWarden"
      },
      {
        "followers": 1923,
        "score": "-0.429771",
        "sentiment": "negative",
        "tweet": "RT @waffen2112: Environmentalist should be called what they really are, subversionist! Another affliction western countries have to deal w\u0026",
        "user": "waffen2112"
      }
    ]
  }
]
```

Figure 11: JSON data format

10 Algorithms

This project does not have many complex algorithms. This is because the Watson Sentiments Analysis API from IBM solves our complexity. The Comparative Analysis function that we are providing uses a simple algorithm that takes two sentiment scores and compares them, informing the user of how their interested keyword compares to their benchmarked keyword.

11 Solution to Challenges

11.1 Challenges

Following is the challenge addressed in the Problem Statement:

The challenges for this project would be figuring out the way to go through social media and getting all the necessary information needed based on a topic or a keyword. We have to work with the information that is freely available on the internet. One of the major issue with social media is all the information can be biased, so the information we collect has to be presented in a manner that it does not come across as factual. Also, picking what kind of organizations to focus on is a challenge because not all organizations are actively discussed on social media platforms.

11.2 Solution

The first challenge addressed was finding the means through which to go through social media data. This was solved by utilizing the Twitter API, which provides many different ways of retrieving tweeted data. Data is returned in JSON format so not very user friendly way of displaying. Therefore, the data will be presented in a systematic and understandable manner.

The second challenge addressed was the bias involved with the information from social media. Because of this reason, the amount of data to be collected in order to do the analysis will be in big sizes in order to have lesser impact from biases.

The third challenge addressed is what kind of organizations to focus on. This was solved by leaving the choices of organizations or institutions to be openly decided by the end user.