**Introduction**

The following is a tutorial on 'how an individual can leverage data available over Twitter' and the kinds of analysis that we could perform such as mood classification, sentimenet analysis on a subject/discussion, geospatial analysis and other forms of intuitive visualisations for better communicating results to stakeholders. This is a growing feild of study and research by policy, psychology and science students.

There are several parts to this tutorial but largely we could divide them into three parts:

* Access Twitter API
  + Authentication (oauth2)
  + Pagination
  + Rate Limiting
* Simple Sentiment Analysis (using TextBlob)
* Geo-Spatial Analysis
* Download data [from my google drive (68 mb)](https://drive.google.com/open?id=10s9Ldj_gawj_DC5UZhWCbebjPeK2KqXW)

We will explore the varrious methods we could use to extract data from 'Twitter' and discuss the Pros & Cons of each.

(**Note:** references used to develp code chunks would be provided above the chunk along with description)

**Authentication**

Most large web scale companies use a combination of authentication and rate limiting to control access to their data to ensure that everyone using it abides. Twitter uses the oauth2 authentication and hence we will go forward to apply for a developer account.

**Twitter API Access**

1. Create a Twitter Account (if you dont have one already)
2. Go [here](https://apps.twitter.com/app/new) and log in.
3. In 'Application Management' - 'Create an Application'. Enter 'Name', 'Description' & 'Website' (can be dummy)
4. Accept the Terms of Service and Create the application
5. Go to 'Keys and Access Tokens'. Scroll down and select 'Create my Access Tokens' to get the Access key and Secret as well

* Save the keys in four files one for each (as whoen below) and access them using the below function:

**witter API**

We will first explore 'Twitter' by simply exploring the **request** library. The procedure is as mentoined below:

* First 'API key & secret' and encode in the ASCII format
* Use the **base64** library to encode the the key.
* Authorization in the 'headers' parameters where we pass this encoded key (along with some other parameters).
* Check for Status (200 OK) and keys that we can use from data response (access\_token).
* Extract the tweets
* Our observations from 'studying the tweets'

Source:

[References 1](https://speca.io/speca/twitter-rest-api-v1.1)

(**Note:** This is a simple User Authentication that permits 180 requests per 15 mins and maximum 100 tweets per each)

### Lets Extract tweets:

* Simply re-use the tokens access tokens as in above. Here the **json** library can be put to work.
* Create a 'Search Parameter' with the following keys:
  + 'search query' as 'facebook'
  + 'language' as 'en' (ISO format)
  + 'result' as 'popular' (can be 'mixed' or 'recent' as well)
  + 'count of tweets' as 5 (to sample, maximum can be 100)

### Observation from exploring tweets

We can observe the following:

* Several tweets are not being able to process and are getting cut-out from the text
* Once we save this text on-to a csv like file several white spaces (tabs, newlines, etc.) will ruin the formatting of our table
* There is alot of noise in our text like emoticons, links, hastags, slangs etc.
* There are a few caveats to pagination, rate limiting and ultimately authentication (user & application) that make a difference to the way we approach the problem

## Access Twitter using 'Tweepy'

As we have discussed above 'user authentication' has its own limitations. "Tweepy" is a well known twitter library that has been developed which will increase the request to 450 per 15 minuites. It gives fexibillity to user to better manage **pagination** & **rate limiting** (wait\_on\_rate\_limit & wait\_on\_rate\_limit\_notify). It returns a simple json, that can be easily parsed and information can be extracted.

We would like to attempt to access twitter using 'tweepy' and extract a small sample of 5 tweets to analyse its working. Also, get the 'full text' of the tweet and not use incomplete tweets going forward.

### Observation from exploring tweets

We can observe the following:

* Manage 'retweeted tweets',as they are incomplete even after using (tweet\_mode='extended')
* We still need to clean the text for our csvfile (tabs, newlines, etc., which might ruin the formatting of our table)
* There is still alot of noise in our text like links, hastags, punctuations etc.

Lets create the following:

* Cleaning tweets function
* Code that maintain retweets as well and saves everything (along with screen name, coordinates and location) to a csv file

## Run the below test cases for tweets (cleaning them) Tweepy 'Cursor' versus 'Search'

Tweepy has three functionalities to extract data:

* Cursor
* Search
* Stream Listner

### Tweepy 'Cursor'

The cursor is a newly implemented functionality and so I wanted to try it out once with a small sample of tweets to search for.

Apply the **retweeted check** and accordingly display the full text

Apply **cleaning** before displaying the tweet

## Start using Twitter to store in CSV

### Tweepy 'Search'

Although, **cursor** is a newer functionality and needs much lesser code. I will still go ahead and use the **search** functionality. We can see from our observation that the coordinates are very hard to find and require additional checks before they can be stored. in a csv file. Additionally we will also record and use the below parts in the code:

* Define the 'maximum number of tweets' that we wish to download
* Keep the 'number of tweets returned per query' constant (i.e. 100)
* Use **max\_id** & **since\_id** to keep a record of donwnloaded tweets and avoid any duplication
* Keep a delay timer, so as to avoid rate limit
* store the query in the csv file with the additional checks of **retweet\_status** & **coordinates**
* "tweet -- coordinates -- coordinates"
* Search for 1 Million Tweets (as max number of rows of a csv file is 1,048,576)

Source:

* [Reference 1](https://www.karambelkar.info/2015/01/how-to-use-twitters-search-rest-api-most-effectively./)
* [Reference 2](http://www.dealingdata.net/2016/07/23/PoGo-Series-Tweepy/)

**Note on Since\_ID:** If results from a specific ID onwards are reqd, set since\_id to that ID. Else default to no lower limit, go as far back as API allows.

**Note on Max\_ID:** If results only below a specific ID are, set max\_id to that ID. Else default to no upper limit, start from the most recent tweet matching the search query.

# The Coordinates problem

Coordinates are extreemly hard to find for the tweets as the users have not shared or geo-tagged their tweets hence we need to take another feild that would be better in terms of obtaining the latitude and Longitude. In the above chunk of code we are recording the coordinates of the geo-tagged tweets, where the user tags the tweet by enabling and sharing the location. We will continue to record these coordinates but as our **secondary lat & long**.

In the below chuck we are going to record those tweets where the users have enabled their location, which can be seen by:

"tweet -- place -- bounding box -- coordinates"

This gives us a bounding box, where we will apply a calculation to get the center and save the coordinates of those as our **primary lat & long**.

* Total number of tweets found related to our search terms : **362,753**
* Total number of primary coordinates found: **1195**
* Total number of secondary coordinates found: **50**

Hence, the decision to select the 'bounding box' feild was a much better one than simple 'coordinates'.

## Using Classifier for Sentiment Analysis

Sentiment Analysis requires a dataset to "train" and test. We can perform the same using Python's 'textblob' library. TextBlob contains a labeled dataset with movie reviews, where the labels are 'positive' or 'negative'. We will try out different classifiers to classify the tweets from the users. Ultimately we should be able to associate the users having positive or negative views on the topics that we have searched.

## **Note:** Apologies, due to lack of time I have been unable to merge the sentiment analysis with geo-spatial analysis. Also, I had initially planned to create my own NLP processor for the sentiment analysis but again had to use an already existing one (other caveats like managing stop words in the tweets etc. has not been performed). Geo-Spatial Analysis of the tweets

Even after collecting information from different sources within the JSON string accessing the Twitter API we were unable to track location of only 3.3 % of the tweets downloaded. This problem has been documented several places where researchers studying people behaviour have to take measured steps to sample the data available.

We are going to read from our data in the final tweets file and only plot those instances of tweets for which we have the coordinates.

* Standard Library for Geo-Spatial analysis
  + [mpl\_toolkits.basemap](https://matplotlib.org/basemap/users/installing.html)

First we need to clean and get rid of the 'NAN' values and use the 1195 observations (tweets) which have a location, the dataframe will look something like below and we will leverage the **pandas read\_csv** for the same.

Now we need to convert the pandas dataframe into a list for the Latitude and Longitude. This is a requirement as the basemap uses list as data structures for its values. The following steps in the process are fairly simple:

* Define the minimum and maximum lattude and longitude for the boundaries (with a margin if required)
* create a Basemap:
  + Center of the map
  + Resolution = full or high (others options like crude, low, intermediate)
  + Projection as 'Mercator' [other projections](https://matplotlib.org/basemap/users/mapsetup.html)
* Draw Coastlines, Countries, States, Map boundaries and fill with colors in the map
* Converting (lat & lon) to map projection coordinates
* Plot the figure of a certain size by passing the parameters along with lat, long

Source:

* [Reference 1](https://stackoverflow.com/questions/44488167/plotting-lat-long-points-using-basemap)
* [Reference 2](http://www.datadependence.com/2016/06/creating-map-visualisations-in-python/)

### Will capture the below image:

We can clearly see high activity in countries affect by this event of cmabridge analytics namely **United States**, **UK**, **India**, **Singapore** and other locations as well.

**Future Work**

Major functions that are needed to be included:

* Create a geo-coder for the 'location' feild that we have saved (if we can get more coordinates to plot from users account location)
* Create our own text classifier
* Merge sentiment analysis with geo-spatial analysis