Be able to be connected to the Stanford Sentiment Corpus, this one was chosen as, in accordance with the literature, it is the largest publicly available twitter dataset with 1,6 million tweets. It is however not the most accurately sorted as it is all sorted automatically. It also has a scaling system where the results are not strictly binary. This is good for our purposes as this is what I wanted to do.

The results will be looked at and compared to our results. They have been collected from 2009 twitter, which means potentially the differences between the two datasets are: difference in slang that might make results less accurate. Also difference in tweet length, the tweets changed from max 180 to 260 in 2017, while TechCrunch says that that didn’t really change the length of tweets they looked at it one year later and did state that it might be that people are still used to tweeting as they used to [1].

There is also another study that says that it barely changed the length of what people were tweeting, but it was performed on Dutch tweets and was performed only two weeks after the change in character limits [2]. So is probably not super accurate.

The Stanford sentiment dataset is sorted automatically and assumes that any tweet with 😊 is positive and any tweet with ☹ is negative. This is obviously flawed. However it doesn’t actually matter that much as the algorithm used can just use the data and remove the column that sorts the tweets.

This dataset will be used for the formatting of ours, will use the same type of formatting so that both can be compared.

The dataset wasn’t necessarily completely objective on which tweets selected as it used specific queries to collect the tweets. The paper about the dataset can be found here [3].

The different fields on the csv: the polarity of the tweet (0=negative, 2=neutral, 4=positive), id of the tweet, date of the tweet, the query, the user, the text of the tweet.

While only the text data will be used to classify the tweets, the other fields will be included in order to be able to connect the dataset built to the Stanford one.

**Does it matter that the username is in the data?**

Is hybrid sentiment analysis machine learning techniques that you use lexicon method to sort tweets first into sentiments, then can be evaluated by me, and given scores, and then put through the machine learning algorithm.

API key secret: lQF726OQxRqvFoGfUFEBgPSrV4vH8phzRhrmMt2WbHfuVfX4c4

API Key: HM2L20WSoeiDOGuBpWMBhO35Y

Access token: 1604068859706855424-yHTZzJEtlJJbCYiv21RwfxMPmhTRYI

Access token secret: WMwVZ2V7I3sNEQqmISUVy3BrczkCMyfo6oLaDlO6VWRXO

Bearer token: AAAAAAAAAAAAAAAAAAAAABI%2BkgEAAAAAXWj%2B%2Bf6FxPZ9fu9kE1wu0NPZiQM%3DauGTgGLIg4RuJ0rj54Z5I5vVFFHOOm7b4tu21dopElGbR6GhFK

Start with twarc2, I have the keys and the app on twitter, then downloading python 3, and instal twarc using pip on command control.

Use twarc to authorize app to access my account made for this.

Graphical user interface, text

Description automatically generated

There you go.

# Steps

**Step 1 -** Collect Tweets

* Identify how I want that to be done, what requests to send to the API
* Identify how to format properly
* Where to find the json or csv file afterwards

**Step 2 –** Build Dataset

* Figure out what data is collected when tweets are collected
* Data pre-process by removing unnecessary columns, only keeping the ones that are on the Stanford Sentiment Corpus
* Set aside all the data that is on the sentiment corpus that is not strictly tweet content
* Vectorize tweets using TF-IDF
* Use Easy data augmentation techniques
* Figure out how to correctly store a dataset

**Step 3 –** build lexicon dictionary

* Figure out how to build lexicon dictionary
* Figure out which words I want to use
* Figure out how it all works
* Figure out how to assign a number to the tweets

**Step 4 –** Put dataset through lexicon

* Put all the tweets through the lexicon to assign them temporary sentiment assignment
* Go through the tweets afterwards to manually review the sentiment assigned

**Step 5 –** Put through the machine learning algorithms

* Try a couple different machine learning algorithms to determine which one is the most efficient
* Select the best one

**Step 6 –** Build GUI

* Design GUI and implement it
* Have it accept user input
* Figure out how to connect the frontend of the application to the backend
* Take user input of keys and API call and select their tweets to run through the trained machine learning algorithm

Twarc has plugins that can convert line oriented json to csv, which is what we will want.

To start with twarc, create application on API and attach it to project on Twitter Developer Portal.

Can use archive and start-time just like a regular search command and can end up with full archive of all tweets for first day of 2020

twarc2 searches --archive --start-time 2020-01-01 --end-time 2020-01-02 animals.txt animals.json

Maybe if I remove the json stuff it will collect every tweet from that day

Stream command: collect tweets as they happen.

**USE PY -M FOR ACCESS TO ANYTHING PYTHON**

Downloaded the csv plugin for twarc using py -m pip install twarc-csv command in visual studio code.

This is so that the data collected can be automatically saved to a csv.

**Use 3-legged Oath for authenticating other users?**

WERE USING TWEEPY NOW

Using streaming HTTP protocol which makes it so that there is only one connection and every time a match is found it goes through that connection instead of opening a new one each time.

Find how to stream tweets in real time through this [4]. By default what I will get back is the id of each tweet and the text of each tweet but to make it comparable to the Stanford sentiment corpus I need to add the date, the query, and the user.

the 1% random sample of public Tweets provided by the sampled stream endpoint can meet this need since it provides a small subset of data relative to the total amount of public Tweets. Additionally, the data is sent to you in real time as it happens, which will meet the requirement of the data being current.

Install pandas so that we can then collect data into csv files.

Create config file to save all tokens and stuff in it meaning that I can share my files without people seeing all the keys, keeps it safer [5].

The project requirements say that the data must be left to only the raw tweets in the dataset, however since then the project shifted slightly to be able to compare our results with other public datasets meaning that they will have more columns than originally planned, ironically every single one that I used as an example to be removed should be in actually.

Once we have the dataset on the same lines and with blank spaces removed, URLs are taken out. Then VADER is installed and used for the emoticons and for the acronyms. Install vaderSeniment in command line.

Creating word abbreviation dictionary using the list from this website [6]

Built the dictionary, then realized that since we had replaced all punctuation including ‘ with a space previously there were some ‘y’s just chilling from y’all turned into why all. So decided to expand acronyms to help preserve the integrity of the text and to help with the meaning of the sentence.

Used this guy’s slightly modified dictionary [7]

# References

[1] <https://techcrunch.com/2018/10/30/twitters-doubling-of-character-count-from-140-to-280-had-little-impact-on-length-of-tweets/?guce_referrer=aHR0cHM6Ly93d3cuc3RhcnRwYWdlLmNvbS8&guce_referrer_sig=AQAAANdNcmjwbwX0KyhrqCCoj3ZQhEQf2GdKsqXkEPlUVeRn1zivikMJHBGPkQLigA93kKehD7eVNAWWc8_KskonpGhy4H3sbPPWe94Yh2-cux1ezFw7_cZEJcGwdFSJP2n7hUiHSG0QV_tD71b6hwh23lxGXuTTPmAohOvLvxa5agzC&guccounter=2>

[2] <https://www.nature.com/articles/s41599-019-0280-3>

[3] <https://www-cs-faculty.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf>

[4] <https://developer.twitter.com/en/docs/tutorials/stream-tweets-in-real-time>

[5] <https://www.youtube.com/watch?v=Lu1nskBkPJU>

[6] <https://messente.com/blog/text-abbreviations>

[7] https://stackoverflow.com/questions/19790188/expanding-english-language-contractions-in-python