# DATA 698 Data Collection and Analysis

**Topic: Student Counselling Software**

## Data Collection and Analysis

**Irene Jacob**

## I. Fetching data from twitter

Twitter is collection of large datasets. For performing the sentiment analysis on twitter data, the data extraction processing is important. Compared to the other networking sites twitter allows users to share their views openly. With the help of twitter API, twitter gives the expansive access to tweets.

The tweets are collected using Twitter's streaming API, or any other mining tool (for example WEKA), for the desired timeframe of analysis. The twitter app is created by twitter to access the twitter developer account which has same username and password. Using these credentials, the string form of tweets can be obtained from API. [1]

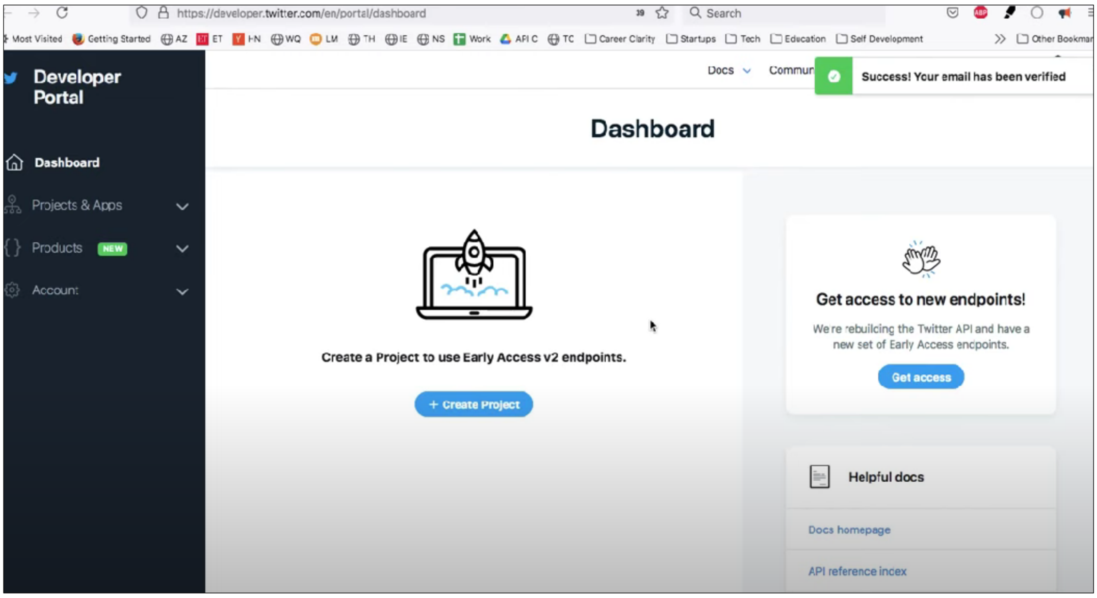
The format of the retrieved text is converted as per convenience (for example JSON). The dataset collected influences the efficiency of the model. The division of dataset into training and testing sets is also a deciding factor for the efficiency of the model. The results depend on the training set. [3]

## II. Searching for Tweets with Python

### Step 1: Set up your twitter developer account

Firstly, Twitter developer account needs to be set up. The below steps can be used to create such an account.

1. Navigate to https://developer.twitter.com/en/apply-for-access and sign in with your twitter account if you already have one. If not, sign up for Twitter with a new account.
2. Click on "Apply for a developer account". This opens a dialogue where you will be asked how you want to use the Twitter API.
3. Describe your intended use: Subsequently, you will be forwarded to a page where you have to state your intended use of your work with the Twitter API. In total, you need to write about 200–600 characters depending on what you intend to do.
4. Review your access application and read the terms.
5. Set up a project and application: Navigate to https://developer.twitter.com/en/portal/dashboard where you can set up your Twitter API project and an application.
6. Copy your bearer token of the application — you will be needing this token shortly. If you can't remember your Bearer Token, navigate to "Keys and tokens" and click on regenerate.



https://developer.twitter.com/en/portal/dashboard

Success! Your email has been verified

Create a Project to use Early Access v2 endpoints.

Get access to new endpoints!

We're rebuilding the Twitter API and have a new set of Early Access endpoints.

**Step 2: Python environment setup**

In order to send API requests in Python and to be able to use the Twitter API I am not going to rely on any Twitter wrapper modules but only on the very handy requests module which can be installed via pip:

pip install requests

**Step 3: Prepare Search Twitter Function**

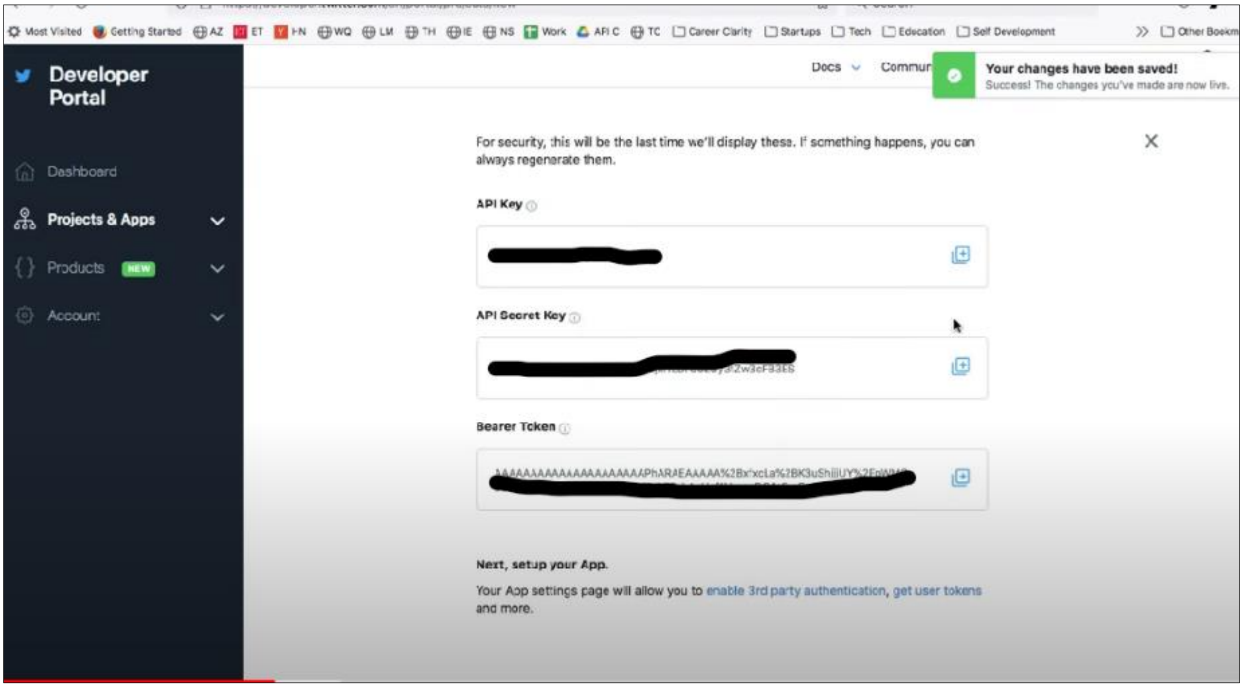
The function is short and needs three parameters:

1. bearer_token [str]: the bearer token copied in Step 1.
2. query [str]: This is the actual string that will be used for matching the desired Tweets. These query strings can be simple such as “family issues”, “exam stress” or very complex and powerful. They allow you to filter tweets by hashtags, identify retweets, exclude certain words or phrases, or only include tweets of a specific language.
3. tweet_fields [str]: Fields to return in the query, such as attachments, author_id, text, etc.
   eg) tweet.fields=text,author_id,created_at

Optional parameters are as follows:

1. max_results: parameters that specify how many tweets should be returned. By default, a request response will return 10 results.
2. start_time: The oldest UTC timestamp (from most recent seven days) from which the Tweets will be provided. (YYYY-MM-DDTHH:mm:ssZ (ISO 8601/RFC 3339).e
3. nd_time: The newest, most recent UTC timestamp to which the Tweets will be provided. (YYYY-MM-DDTHH:mm:ssZ (ISO 8601/RFC 3339).

**Step 4: Run search_twitter function**



Your changes have been saved!
Success! The changes you've made are now live.

For security, this will be the last time we'll display these. If something happens, you can always regenerate them.

API Key

[redacted]

API Secret Key

[redacted]

Bearer Token

[redacted]

Next, setup your App.

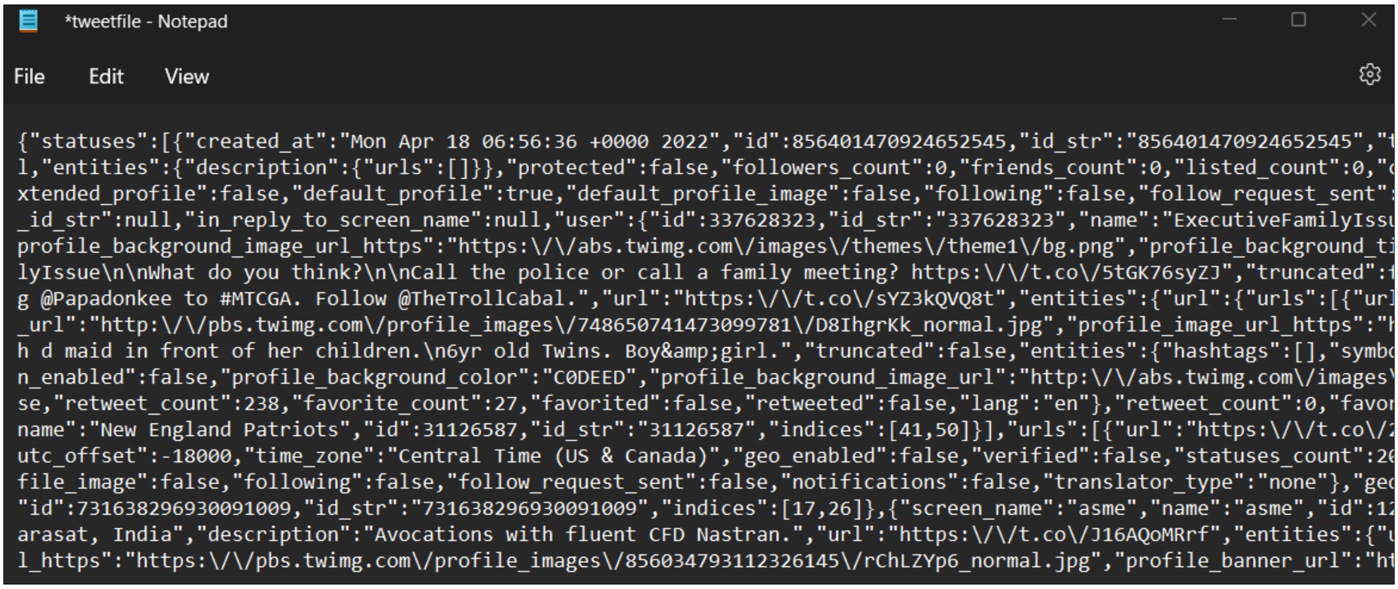
Your App settings page will allow you to enable 3rd party authentication, get user tokens and more.

Manually extract data from twitter using some API available for twitter. For this I have chosen tweepy as an API for extraction of tweets. Tweepy is not compatible with the new versions of python. So, for using this API an older version of python is needed (python 2.7). To access tweets on twitter using API first I had to authenticate the console from which I will be accessing twitter. This could be done by following steps listed below:

1. Creation of a twitter account.
2. Logging in at the developer portal of twitter.
3. Select "New App" at developer portal.
4. A form for creation of new app appears

5. After this the app for which the form was filled out will go for review by twitter team.
6. Once the review is complete and the registered app is authorized then the user is provided with ‘API key’ and ‘API secret’
7. After this "Access token" and "Access token secret" are given.

These keys and tokens are unique for each user and only with the help of these can one access the tweets directly from twitter. [5]

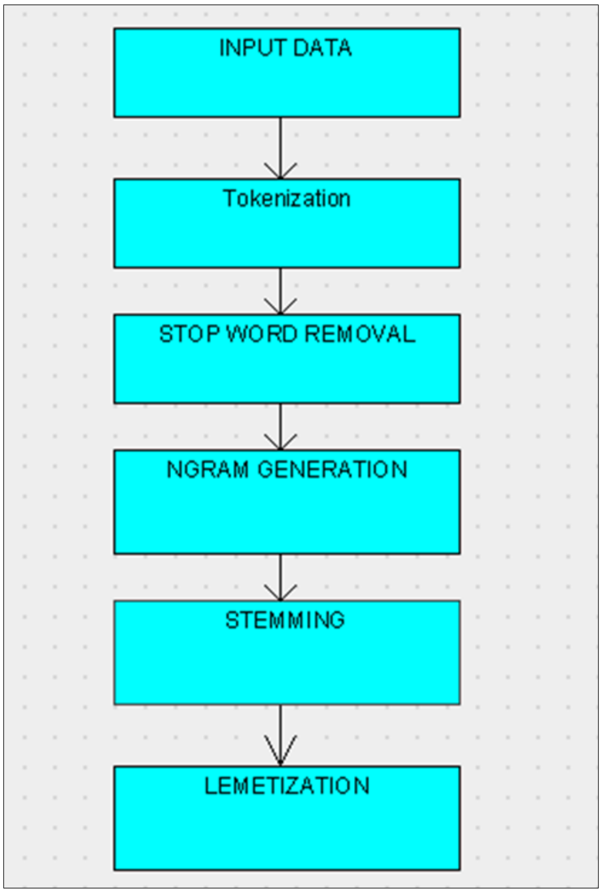


```
{"statuses":[{"created_at":"Mon Apr 18 06:56:36 +0000 2022","id":856401470924652545,"id_str":"856401470924652545","t
l,"entities":{"description":{"urls":[]}},"protected":false,"followers_count":0,"friends_count":0,"listed_count":0,"c
xtended_profile":false,"default_profile":true,"default_profile_image":false,"following":false,"follow_request_sent":
_id_str":null,"in_reply_to_screen_name":null,"user":{"id":337628323,"id_str":"337628323","name":"ExecutiveFamilyIssu
profile_background_image_url_https":"https:\/\/abs.twimg.com\/images\/themes\/theme1\/bg.png","profile_background_ti
lyIssue\n\nWhat do you think?\n\nCall the police or call a family meeting? https:\/\/t.co\/5tGK76syZJ","truncated":f
g @Papadonkee to #MTCGA. Follow @TheTrollCabal.","url":"https:\/\/t.co\/sYZ3kQVQ8t","entities":{"url":{"urls":[{"url
_url":"http:\/\/pbs.twimg.com\/profile_images\/748650741473099781\/D8IhgrKk_normal.jpg","profile_image_url_https":"h
h d maid in front of her children.\n6yr old Twins. Boy&amp;girl.","truncated":false,"entities":{"hashtags":[],"symbo
n_enabled":false,"profile_background_color":"C0DEED","profile_background_image_url":"http:\/\/abs.twimg.com\/images\
se,"retweet_count":238,"favorite_count":27,"favorited":false,"retweeted":false,"lang":"en"},"retweet_count":0,"favor
name":"New England Patriots","id":31126587,"id_str":"31126587","indices":[41,50]}],"urls":[{"url":"https:\/\/t.co\/2
utc_offset":-18000,"time_zone":"Central Time (US & Canada)","geo_enabled":false,"verified":false,"statuses_count":20
file_image":false,"following":false,"follow_request_sent":false,"notifications":false,"translator_type":"none"},"geo
"id":731638296930091009,"id_str":"731638296930091009","indices":[17,26]},{"screen_name":"asme","name":"asme","id":12
arasat, India","description":"Avocations with fluent CFD Nastran.","url":"https:\/\/t.co\/J16AQoMRrf","entities":{"u
l_https":"https:\/\/pbs.twimg.com\/profile_images\/856034793112326145\/rChLZYp6_normal.jpg","profile_banner_url":"ht
```

To obtain the tweets relevant to the data collection of this project, the following URL is used from Twitter API:

***https://api.twitter.com/1.1/search/tweets.json?q=studentlife***

## III. Data Pre-processing



### 1. Tokenization

Input data for training is collected from twitter and is stored in the input file. Tokenization is used to split a stream of text into smaller units called tokens (words or phrases). The tokenization is based on regular expressions (regexp). Some types of tokens (eg. phone numbers or chemical names) will not be captured and will be probably broken into several tokens. To overcome this problem, as well as to improve the richness of the pre-processing pipeline, we can improve the regular expressions. The core component of the tokenizer is the regex_str variable, which is a list of possible patterns. The tokenize() function catches all the tokens in a string and returns them as a list. This function is used within preprocess(), which is used as a pre-processing chain.[2]

### 2. Removing stop words

One of the major forms of pre-processing is going to be filtering out useless data. In natural language processing, useless words are referred to as stop words. It is not desirable to have these words taking up space in our database or taking up processing time. Stop words as words that just contain no meaning, and we want to remove them. This can be done by storing a list of stop words. A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.

### 3. N Gram Generation

Language has a sequential nature, hence the order in which words appear in the text matters a lot. This feature allows us to understand the context of a sentence even if there are some words missing. Gram generation is the process of combining words so that they make more sense. When two words are joined together it is called 2gram generation. I am using 2gram generation for this project.[8]

The formal definition of N Gram is “a contiguous sequence of n items from a given sample of text”. The main idea is that given any text, we can split it into a list of unigrams (1-gram), bigrams (2-gram), trigrams (3-gram) etc. Consider the below example:

Text: “I went biking”

Unigrams: [(I), (went), (biking)]

Bigrams: [(I, went), (went, biking)]

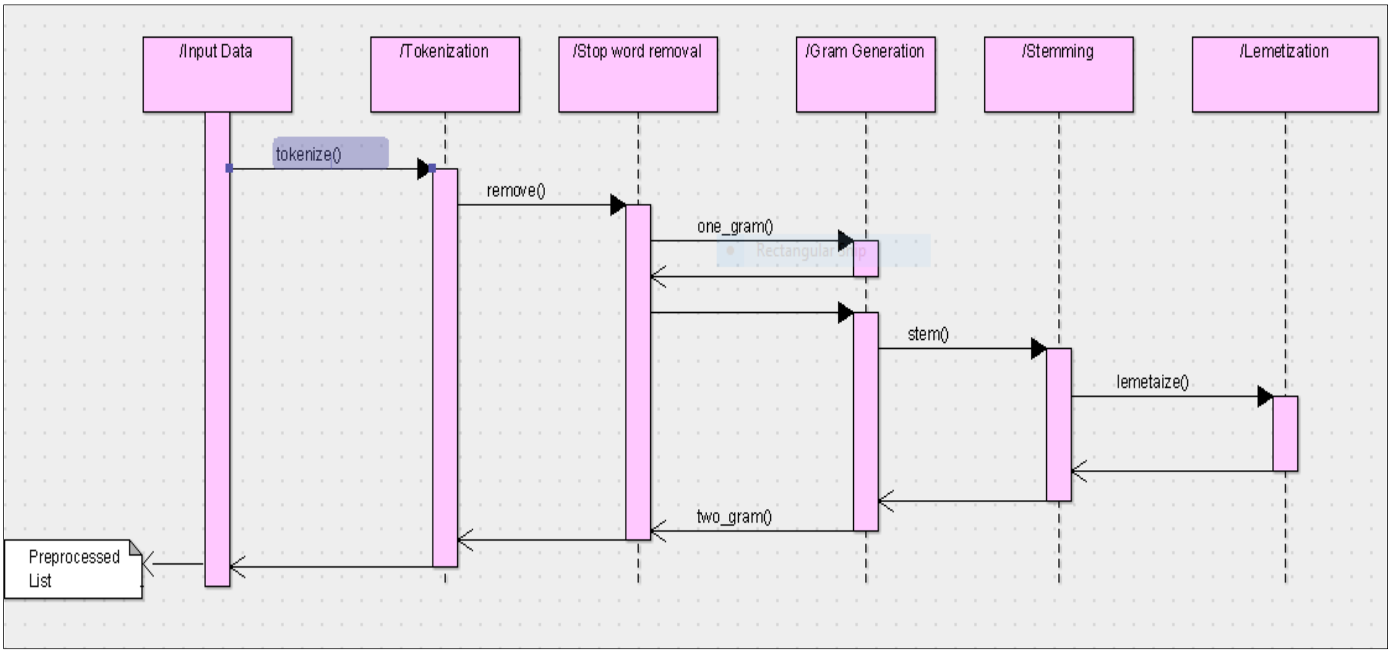
### 4. Stemming

Stemming is considered to be the more crude/brute-force approach to normalization (although this doesn’t necessarily mean that it will perform worse). There are several algorithms, but in general they all use basic rules to chop off the ends of words. Many variations of words carry the same meaning. The reason why we stem is to shorten the lookup and normalize sentences. For example: ride and riding, are words with same meaning. Having individual dictionary entries per version would be highly redundant and inefficient. One of the algorithms is The Porter stemmer algorithm. [7]

### 5. Lemmatization

The goal is same as with stemming, but stemming a word sometimes loses the actual meaning of the word. Lemmatization usually refers to doing things properly using vocabulary and morphological analysis of words. It returns the base or dictionary form of a word, also known as the lemma. Stemming can often create non-existent words, whereas lemmas are actual words.

## IV. Architecture Diagram



**Using Collected Data - Tweets**

The sentiment model needs to be trained against examples of the type of data that I expect to see when I use my model. Some examples of tweets are as follows:

1. "Feeling kind of low...."
2. "OMG! Just had a fabulous day!"
3. "Eating eggplant. Why bother?"

All the samples should be in the same language (though it doesn't matter what language, as long as it’s consistent and space-delimited).

**Labelling the Data**

Once the training samples have been collected, I pre-classify each sample with a label. A label is a string that best describes that example, for example: "happy", "sad", "on the fence". So, to assign labels to the previous examples:

1. "sad", "Feeling kind of low...."
2. "excited", "OMG! Just had a fabulous day!"
3. "bored", "Eating eggplant. Why bother?"

A model can have up to 1000 labels, but the best practise is to use as many labels as required for the problem at hand. Each label should have at least a few dozen examples assigned to it. Labels are just strings, so they can have spaces. However, double quotes should be put around any labels that have spaces, and any nested quotation marks should be escaped using a \ mark. Example: "that\'s fine" Labels are case-sensitive. So "Happy" and "happy" will be seen as two separate labels by the training system. Best practice is to use lowercase for all labels, to avoid mix-ups. Each line can only have one label assigned, but multiple labels can be applied to one example by repeating an example and applying different labels to each one. For example:

1. "excited", "OMG! Just had a fabulous day!"
2. "annoying", "OMG! Just had a fabulous day!"

### Preparing the collected data

The data is formatted as a comma-separated values (CSV) file with one row per example. The format of this file is basically this:

label1, feature1, feature2, feature3, ....

label2, feature1, feature2, feature3, ....

So, the file would look something like this:

"sad", "Feeling kind of low...."

"excited", "OMG! Just had a fabulous day!"

"bored", "Eating eggplant. Why bother?"

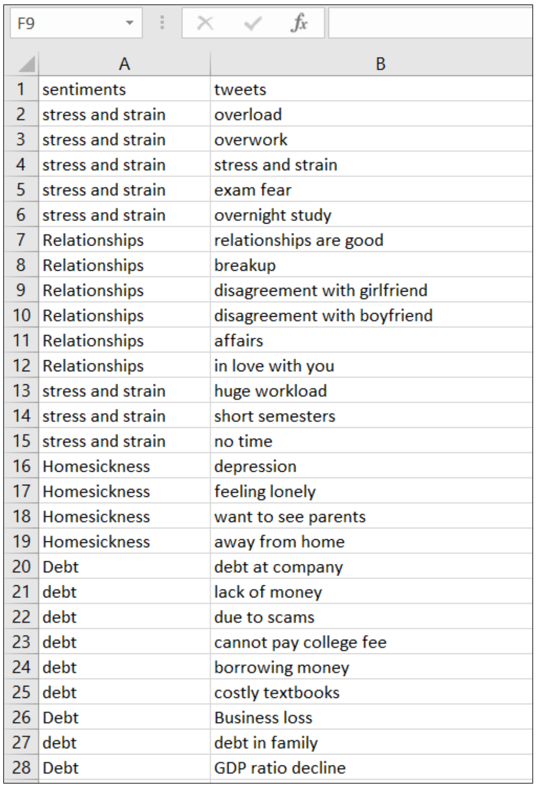
### Training the Model with the Naïve Bayes Classifier

The training of the Naive Bayes Classifier is done by iterating through all the documents in the training set. From all the documents, a Hash table with the relative occurrence of each word per class is constructed.

This is done in two steps:

1. Construct a huge list of all occurring words per class:
   ```
   for ii in range(0,len(Y)):
   label = Y[ii]
   self.nb_dict[label] += X[ii]
   ```

2. calculate the relative occurrence of each word in this huge list, with the "calculate_relative_occurences" method. This method uses Python's Counter module to count how much each word occurs and then divides this number with the total number of words. The result is saved in the dictionary nb_dict.



| | A | B |
|---|---|---|
| 1 | sentiments | tweets |
| 2 | stress and strain | overload |
| 3 | stress and strain | overwork |
| 4 | stress and strain | stress and strain |
| 5 | stress and strain | exam fear |
| 6 | stress and strain | overnight study |
| 7 | Relationships | relationships are good |
| 8 | Relationships | breakup |
| 9 | Relationships | disagreement with girlfriend |
| 10 | Relationships | disagreement with boyfriend |
| 11 | Relationships | affairs |
| 12 | Relationships | in love with you |
| 13 | stress and strain | huge workload |
| 14 | stress and strain | short semesters |
| 15 | stress and strain | no time |
| 16 | Homesickness | depression |
| 17 | Homesickness | feeling lonely |
| 18 | Homesickness | want to see parents |
| 19 | Homesickness | away from home |
| 20 | Debt | debt at company |
| 21 | debt | lack of money |
| 22 | debt | due to scams |
| 23 | debt | cannot pay college fee |
| 24 | debt | borrowing money |
| 25 | debt | costly textbooks |
| 26 | Debt | Business loss |
| 27 | debt | debt in family |
| 28 | Debt | GDP ratio decline |

### Making Predictions and Testing the Prediction Set

Now that the model is successfully built, it is time to make predictions. The output from a prediction call for a classification problem, like sentiment analysis will include several important fields. This dictionary can be updated, saved to file, and loaded back from file. It contains the results of Naive Bayes Classifier training.

Classifying new documents is also done quite easily by calculating the class probability for each class and then selecting the class with the highest probability.

## V. References:

1. Deshapnde, P., Joshi, P., Pawar, P., Madekar, D., & Salunke, P. (2019). *A Survey On: Sentiment Analysis framework of Twitter data Using Classification*. Retrieved 30 March 2022.
2. SIDDHARTH, S., DARSINI, R., & SUJITHRA, D. (2018). *SENTIMENT ANALYSIS ON TWITTER DATA USING MACHINE LEARNING ALGORITHMS IN PYTHON*. Retrieved 30 March 2022.
3. Gupta, B., Negi, M., Vishwakarma, K., Rawat, G., & Badhani, P. (2017). *Study Of Twitter Sentiment Analysis Using Machine Learning Algorithms On Python*. Retrieved 30 March 2022.
4. Khattak, A., Batool, R., Satti, F., Hussain, J., Khan, W., Khan, A., & Hayat, B. (2020). *Tweets Classification And Sentiment Analysis For Personalized Tweets Recommendation*. Retrieved 30 March 2022.
5. Faizan, F. (2019). *Twitter Sentiment Analysis*. Retrieved 30 March 2022.
6. Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011). *Sentiment Analysis Of Twitter Data*. Retrieved 30 March 2022.
7. Alsaeedi, A., & Khan, M. (2019). *A Study On Sentiment Analysis Techniques Of Twitter Data*. Retrieved 30 March 2022.
8. Gamal, D., Alfonse, M., El-Horbarty, E., & Salem, A. (2019). *Implementation Of Machine Learning Algorithms In Arabic Sentiment Analysis Using N-Gram Feature*. Retrieved 30 March 2022.
9. X. Chen, M. Vorvoreanu and K. Madhavan, "Mining Social Media Data for Understanding Students' Learning Experiences," in IEEE Transactions on Learning Technologies, vol. 7, no. 3, pp. 246-259, July-Sept. 2014, doi: 10.1109/TLT.2013.2296520.
10. Walaa Medhat, Ahmed Hassan, Hoda Korashy, Sentiment analysis algorithms and applications: A survey, Ain Shams Engineering Journal, Volume 5, Issue 4, 2014, Pages 1093-1113, ISSN 2090-4479, https://doi.org/10.1016/j.asej.2014.04.011.
11. C. Kaur and A. Sharma, "Social Issues Sentiment Analysis using Python," 2020 5th International Conference on Computing, Communication and Security (ICCCS), 2020, pp. 1-6, doi: 10.1109/ICCCS49678.2020.9277251.
12. Kharde, Vishal & Sonawane, Sheetal. (2016). Sentiment Analysis of Twitter Data: A Survey of Techniques. International Journal of Computer Applications. 139. 5-15. 10.5120/ijca2016908625.