Assignment in Information Retrieval Course

**tools used:**

Main class made in python language the extract all tweets and covert them to text then start working on them using different tools I developed and more tools I imported from online library.

You all need is to run Main class and the file will start working, stages of main class:

1.import data to text file:

I imported each tweet to and array and put all data in text file.

I had trouble saving data in python program file size is too large.

File name which contain all tweets is called text.txt

print('converting excel file to csv \n')  
f= open("text.txt","w+")  
wb = load\_workbook(filename='data.xlsx', read\_only=True)  
ws = wb['sheet']  
iterRows=iter(ws.rows)  
next(iterRows)  
for row in iterRows:  
 i=0  
 for cell in row:  
 i=i+1  
 if i==7:  
 text=str(cell.value)  
 for word in text:  
 f.write(word)  
 f.write('\n')  
f.close()  
wb.close()

2.secondly we started importing each tweet and tokenize the text of that tweet using nltk library

saved tokenized tweet could be found in tokens.txt file.

NLTK-WIKI source:

The **Natural Language Toolkit**, or more commonly **NLTK**, is a suite of [libraries](https://en.wikipedia.org/wiki/Library_(computer_science)) and programs for symbolic and statistical [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing) (NLP) for English written in the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)). It was developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the [University of Pennsylvania](https://en.wikipedia.org/wiki/University_of_Pennsylvania).[[4]](https://en.wikipedia.org/wiki/Natural_Language_Toolkit#cite_note-4) NLTK includes graphical demonstrations and sample data. It is accompanied by a book that explains the underlying concepts behind the language processing tasks supported by the toolkit,[[5]](https://en.wikipedia.org/wiki/Natural_Language_Toolkit#cite_note-Bird-Klein-Loper-5) plus a cookbook.[[6]](https://en.wikipedia.org/wiki/Natural_Language_Toolkit#cite_note-Perkins-6)

print('prepearing tokenization \n')  
f=open("text.txt","r")  
tW= open("Tokens.txt","w+")  
tokens=[]  
for line in f:  
 tLine=repr(nltk.word\_tokenize(line))  
 tW.write(tLine+'\n')  
f.close()  
tW.close()

3.thirdly I imported each word from each tween, Couse I tokenized each tweet separately, using a python method of class string lower.

stopWords=open("Tokens.txt");  
appendFile = open('caseFolding.txt', 'w+')  
for words in stopWords:  
 words = eval(words)  
 for r in words:  
 appendFile.write(r.lower()+' ')  
 appendFile.write('\n')  
appendFile.close()

4.we then imported all stop words that can be found in nltk library

import nltk

{‘ourselves’, ‘hers’, ‘between’, ‘yourself’, ‘but’, ‘again’, ‘there’, ‘about’, ‘once’, ‘during’, ‘out’, ‘very’, ‘having’, ‘with’, ‘they’, ‘own’, ‘an’, ‘be’, ‘some’, ‘for’, ‘do’, ‘its’, ‘yours’, ‘such’, ‘into’, ‘of’, ‘most’, ‘itself’, ‘other’, ‘off’, ‘is’, ‘s’, ‘am’, ‘or’, ‘who’, ‘as’, ‘from’, ‘him’, ‘each’, ‘the’, ‘themselves’, ‘until’, ‘below’, ‘are’, ‘we’, ‘these’, ‘your’, ‘his’, ‘through’, ‘don’, ‘nor’, ‘me’, ‘were’, ‘her’, ‘more’, ‘himself’, ‘this’, ‘down’, ‘should’, ‘our’, ‘their’, ‘while’, ‘above’, ‘both’, ‘up’, ‘to’, ‘ours’, ‘had’, ‘she’, ‘all’, ‘no’, ‘when’, ‘at’, ‘any’, ‘before’, ‘them’, ‘same’, ‘and’, ‘been’, ‘have’, ‘in’, ‘will’, ‘on’, ‘does’, ‘yourselves’, ‘then’, ‘that’, ‘because’, ‘what’, ‘over’, ‘why’, ‘so’, ‘can’, ‘did’, ‘not’, ‘now’, ‘under’, ‘he’, ‘you’, ‘herself’, ‘has’, ‘just’, ‘where’, ‘too’, ‘only’, ‘myself’, ‘which’, ‘those’, ‘i’, ‘after’, ‘few’, ‘whom’, ‘t’, ‘being’, ‘if’, ‘theirs’, ‘my’, ‘against’, ‘a’, ‘by’, ‘doing’, ‘it’, ‘how’, ‘further’, ‘was’, ‘here’, ‘than’}

5. Stemming words, we imported already found library found in the NLTK called porter

from nltk.stem import porter

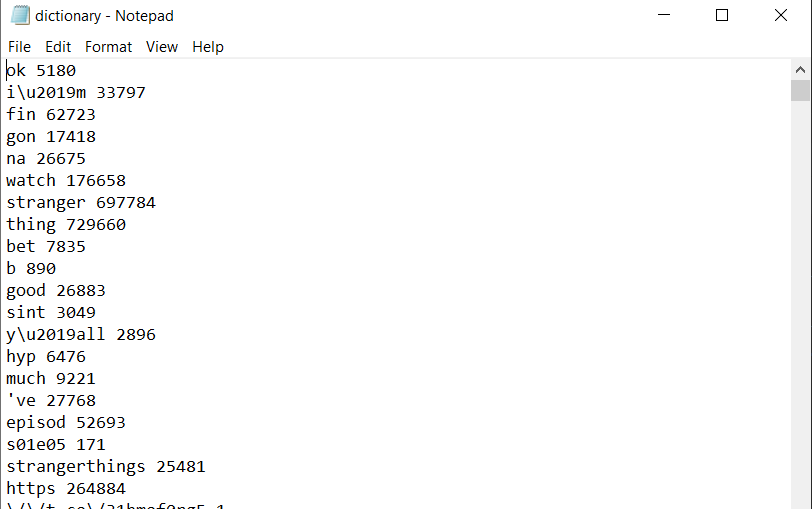
then we used the library to stem each word in each tweet:

print('stemming text\n')  
o=open("stopWordsRemoval.txt");  
appendFile = open('stemming.txt', 'w+')  
ps = porter.PorterStemmer  
for line in o:  
 line=eval(line)  
 for r in line:  
 word=r;  
 word=ps.stem(word)  
 appendFile.write(word+' ')  
 appendFile.write('\n')  
appendFile.close()

6.we then created dictionary for the whole file using basic python methods:

with open('stemming.txt') as f:  
 c = collections.Counter(f.read().split())  
appendFile = open('dictionary.txt', 'w+')  
for word in c:  
 appendFile.write(word+' '+str(c[word])+'\n')  
appendFile.close()

file name ‘dictionary.txt’



7. Sentiment analysis: we started an api that give each words an sentiment score and then we calculate the whole score but there were couple problems Couse some words can go either ways, the TextBlob library is the best we could get for free, there are better non-free library’s.

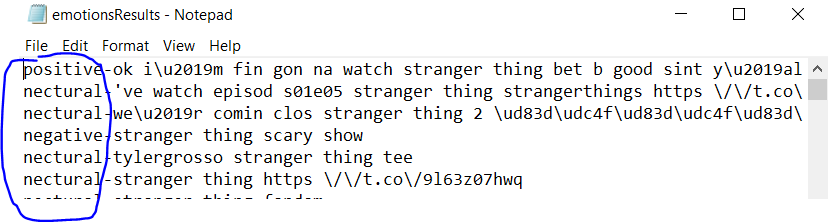
***TextBlob:***

a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

text=open('stemming.txt','r')  
appendFile = open('emotionsResults.txt', 'w+')  
for line in text:  
 blob=TextBlob(line)  
 x=blob.sentiment.polarity;  
 state='nectural'  
 if x!=0:  
 if(x>0):  
 state='positive'  
 else:  
 state='negative'  
 appendFile.write(state+"-"+line)  
text.close()  
appendFile.close()

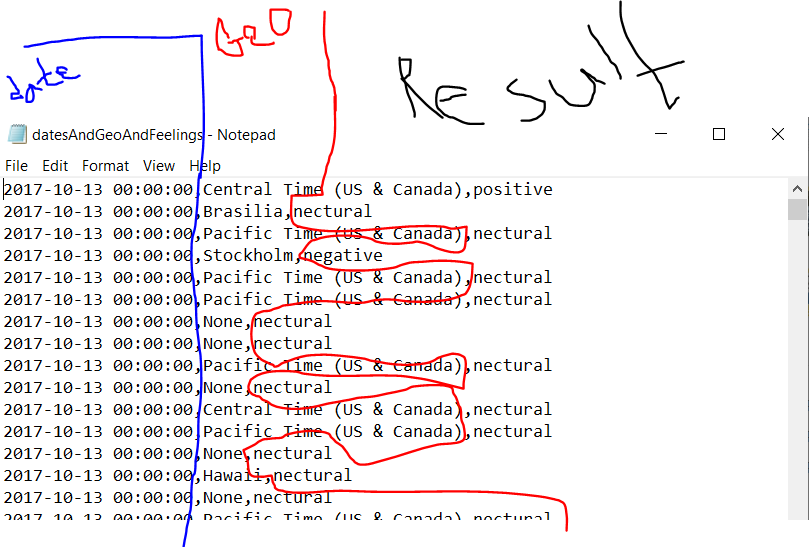
the process is putting each tweet score in a file called ‘emotionResults.txt’ that had each tweet and the total score results which means:

result=total score of words, each word is scored with the TextBlob library.

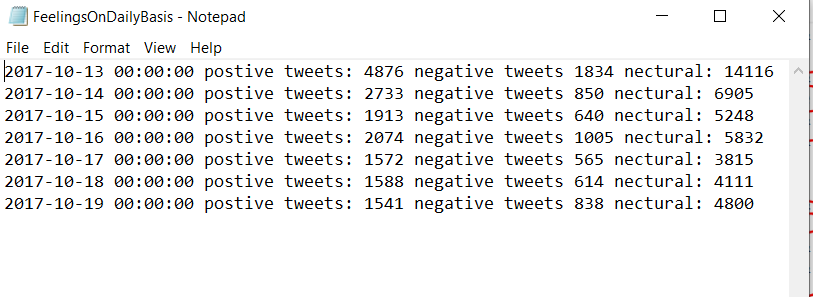


For last processing and graph building I put three elements in one file for non-single tweet information relevant, but a total of tweet’s.

In a file called ‘ datesAndGeoAndFeelings.txt’ I put the score, date, and geo for each tweet.

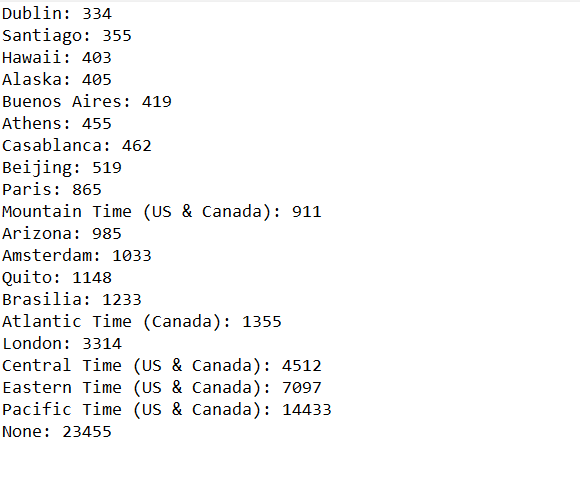


The file is order by date and I counted the total tweet score on a daily basis and put in a file called ‘FeelingOnDailyBasis.txt’:



Graph for description:

From the same file we count positive negative and total of tweet per geological area:



Then we calculate for each type of tweets per geo:

**postive**: {'Arizona': 985, 'Amsterdam': 1033, 'Quito': 1148, 'Brasilia': 1233, 'Atlantic Time (Canada)': 1355, 'London': 3314, 'Central Time (US & Canada)': 4512, 'Eastern Time (US & Canada)': 7097, 'Pacific Time (US & Canada)': 14433, 'None': 23455……}

**negative**:{ 'Quito': 130, 'Atlantic Time (Canada)': 144, 'London': 418, 'Central Time (US & Canada)': 419, 'Eastern Time (US & Canada)': 627, 'Pacific Time (US & Canada)': 1321, 'None': 2157…..}

**nectural**: {'Europe\\/Warsaw': 1, 'America\\/Buenos\_Aires': 1, 'Europe\\/Berlin': 1, 'Ulaan Bataar': 1, 'America\\/Regina': 1, 'Kabul': 1, 'Africa\\/Tunis': 1, 'Europe\\/Oslo': 1,…..}

*these can be found in console in python file(main.py).*

The three location in which most tweeted came from are:

1.us and Canada: 28,308‬

2.London: 3362.

3.Brasil: 1233.

Unknown places came first actually but since that in unanimous and not useful I will ignore it.

Another reason to ignore it is us and Canada exceed that unknow total of tweets.

Positive tweets in us and Canada: 7,718

Positive tweets in London:2031

Positive tweets in brasil:75

Negative tweets in us and Canada: 2,584

Negative tweets in London:422

Negative tweets in brasil:60

Nocturnal tweets in us and Canada:18,006

Nocturnal tweets in London: 909

Nocturnal tweets in brasil:1098

**The following page show the number of each type of tweets in each of the three location of the most places the tweets came from**

The percentage of brasil in every type of tweet was so small we had to add it to another category:

‬

**Word clouds**

This a word cloud for the entire text:

A close up of text on a white background

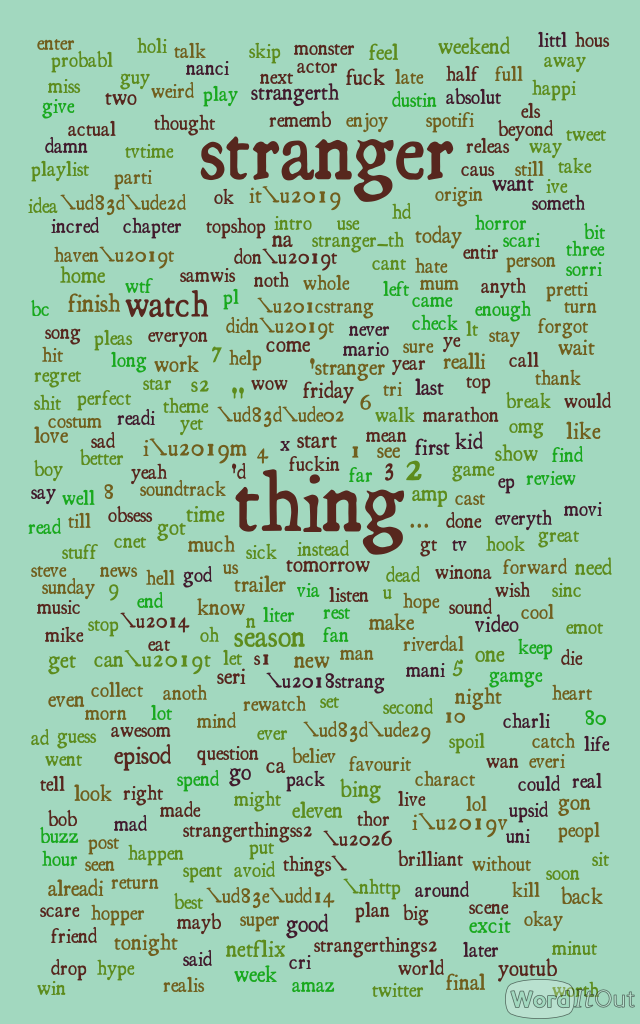
Description automatically generated

This a word cloud for us and American areas:

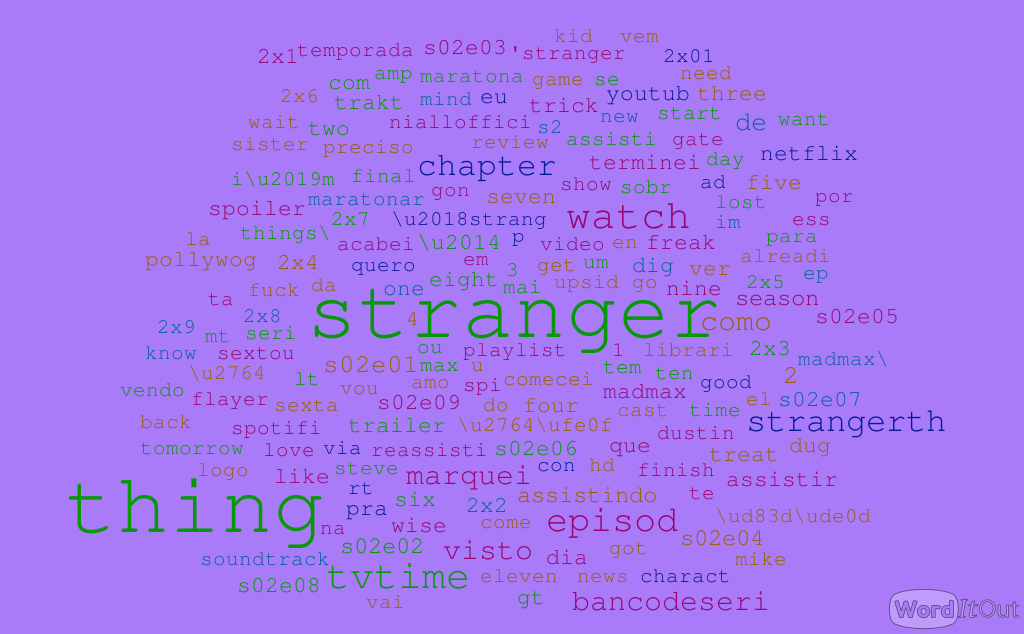
We add all times zones in us and Canadian areas and extract all tweet then we do same process we did on original text:



Secondly geographic place been tweeted from is London so here’s a word cloud of London Tweet’s:



Word cloud of brasil:



World clouds: what we did is we converted each text after doing all the operations o the text itself we convert it to dictionary when we counted each word and then made a word cloud of that dictionary where we made the most frequent word noticeable with bigger font size and different color.

**Tools we used:**

For making dictionary we used library preinstalled in python called collections and we counted each word and put in a dictionary after we did all the text operations on the text.

The we made a word cloud of that dictionary using word cloud site called ‘wordcloud.com’

**Noticeable thing while progressing the text:**

First thing we encounter was the large text, we suggest three ways to solve the problem as we learned

We could work on one pc and enlarge the ram or two pc’s and divide the excel file or simply divide the text to portions and work on each portion alone and merging each portion to the result. We stick with dividing the text to portions cause that was the faster and easiest thing we could do.

In tokens we noticed a lot of wrong pronounce in tweets which caused us a bit of trouble and a spaces inside words but that was very small part of the text and we ignore it.

We used porter stemmer in order to stem words but while searching for solution we encounter the WordNetLemmatizer and SnowballStemmer which were kind of Lancaster Stemmer but they did the correction degree which we learned about and could correct the word if the world had a grammar fault and extract the source of it, plus to the facet that they would ignore irrelevant words like stop words.

**We only analyzed the text in methods that only works in English**, we encounter small percentage of tweets from each country that had the same language of the country, thing that was very hard for to work and detect since some didn’t mention geographical place the tweet came from and same languages had the same letters as in English.

In word profiling we encounter type of problem that we had to analyze the word according to the sentence but we didn’t we give each word a value which actually can go either way, I’m really sorry for the word I’m about to use

The show was shit

The show is the shit

We give the word ‘shit bad score’ but in the library we used it wasn’t high score since it can go either ways.

Results:

In Word cloud of the non-geographic image it was very cleared to the eye couple of words:

The bigger font words were

1.Stranger

2.thing

3.2

4.season

If we actually want to put these words in sentence we can get :

**Stranger Thing season 2**

Another slightly apparent word is watch which imply to screen related tv show or movie not a video game or story.

Since it’s season we can be sure that’s a series.



**Thing we can analyze:**

the word cloud focused on the most shown words in the text:

non-relevant tweet’s in the largest in the group, checking some of the word in word cloud such as ‘download’, ‘watch’,’e1s2’-which imply a certain episode is imply to the episode itself, a way to watch it or commercials.

The second degree words in the word cloud were mostly feeling type like negative words like ‘bad’, ’suck’, ‘shit’ ,’depress’ which almost were equal the amount of good words in the text ‘super’, ’good’, ’wow’, ’proud’ which kind give us mixed feelings about the show.

Thirdly group in word cloud was consists of words of action witch we mostly described something in the tv show’s another thing helped us to be certain is the fact that is this groups was the name of two main characters ‘bobby, ’harry’, ’mille brown’, we can also relate these feeling to actions since some like ‘proud’ has to be connected to an action so we know people actually being emotionally involved in the series.

Another mention was Netflix which is a big platform for these tv series which the show is on.

**The word ‘Friday’ were also apparent which caught our attention:**

The first day in all the tweet is 13/10/2017 which were Friday, now the word Friday also were mentioned in the word cloud, which we can assume the day an episode were streamed, that same day had the most tweets which mostly were neutral, although the amount of positive tweet were smaller they were also large amount and almost doubled the negative tweets for the rest of the week which give us a **sign of good feedback** on that Friday episode .

We noticed that English speakers were more actively and mixed with the tweets expressing how they feel about the episode we had more tweets describing feelings than neutral which we can assume that it was commercials, we can say the language was a factor since brasil had much less interaction. Anther factor we can provide is that people in London are largely into the episode or the second assumption is the lack of commercial in of the show in that area which only caught the feeling interaction of people, we can absolutely assume the opposite in brasil which had more large commercial for the show and the people didn’t interact or that the people simply didn’t like the show.

It was type of obstacle to actually understand the Brazilian word cloud since we didn’t really understand portages unlike the English which we totally understand but we couldn’t really decide between commercially tweets or actually people tweets, but if we had to deiced we will go with an assume that according to chart which show that the most tweets in neutral and positive and negative are people tweets which expressed positive feedback.

//for creating all necessary files please run main file in python

I think you might need to download required libraries for me it was easy I did with pycharm program.

.

**Name: saleem sabeihe 316573898**

**Name :mohammed Abid:**