FIT5196 Task 1 in Assessment 1

Student Name: Gayatri Aniruddha

Student ID: 30945305

Date: 29/08/2020

Version: 1.0

Environment: Python 2.7.11 and Jupyter notebook

Libraries used: please include the main libraries you used in your assignment here, e.g.,:

- re (for regular expression, included in Anaconda Python 2.7)
- langid (to classift the language of a tweet)
- os (to interact with the operating system)

1. Introduction

- Here, we have been provided with a **twitter dataset text files**.
- Our text file contains tweets related to COVID-19.
- We have the following information about the tweets :
  - id, text and created_at dates.
- We need to perform the following tasks on this dataset :
  - We need to extract the data present in the text file.
  - The extracted data must be transformed into an XML format.
- Here, the XML file has id, text and created_at dates.
- We have also ensured the following:
  - We have filtered out the tweets and have kept only the english tweets.
  - We have also ensured that the tweet_id's are unique.
- We have also followed this constraint that only 3 packages have been used.

2. Methodology

- I have followed the following steps in order to get the desired output for task 1.

2.1 Importing Libraries

- This section contains the code to import the libraries we need in this assessment.
- Here, we have imported os, re and langid.
- These are the only packages that we are allowed to use for this task 1.

In [1]:

```
# Library to interact with the operating system
import os

# Library for regular expression
import re

# Library to filter out the non engligh tweets
import langid
```

In [2]:

```
# Importing the required packages from langid package
# These packages are imported for separating out the english tweets

from langid.langid import LanguageIdentifier
from langid.langid import model
```

In [3]:

```
# Reference : https://github.com/saffsd/langid.py
# Initialised the identifier with normal probability True
identifier = LanguageIdentifier.from_modelstring(model, norm_probs = True)
```

2.2 Reading the text files

- Here, we have used the os walk function in order to load the data.
- We have then created a list to store all the tweets.

In [4]:

```
# Initializing a list named list_tweet to store the tweets
# Here, every element of the list is a tweet
list_tweet = []

# Here we access the files stored in our system and store the tweets in a list
for root, dirs, files in os.walk("C:\\Users\\Gayatri Aniruddha\\Desktop\\Sem 2 2020\\FIT519

    # Iterating through every file
    for file in files:
        # join is the path to the file
        with open(os.path.join(root, file), 'r', encoding = 'utf8') as t:

            # Reading the content of the file into tweet
            tweet = t.read()

            # Writing the contents of the text into our list
            list_tweet.append(tweet)
```

2.3 Reading the tweets.

- Here, we have initialised a dictionary to store all the twitter data.
- Dummy Dictionary :

    - This contains words which have to be changed for data cleaning
    - Key is the word that is replaced
    - Value is the word that it is replaced with
    - Here, we have done this to help eval
    - For tweets : we use lower case
- Pattern :
    - We can always combine a regular expression pattern into pattern objects.
    - This is further used for pattern matching.
    - This also helps to search a pattern again without having to re-write it again and again.

2.3 Data Manipulation and Analysis

- We then use the eval function to get our tweets in a dictionary format.
- We then take care of the emojis.
    - Here, emojis are utf-16 encoded.
    - We convert them to utf-8 encoding as utf-8 is the standard format.
- Then, using langid, we only keep the tweets in english.
- Finally, we add the tweet a dictionary according to the specified format.

In [5]:

```
# This contains words which have to be changed for data cleaning
# Key is the word that is replaced
# Value is the word that it is replaced with
# This has been done to facilitate the eval function
dummy_dictionary = {'true': 'True', 'false': 'False'}

# Here we tell the compiler that it is just a variable.
# Do not worry about it and go ahead with the reading
# Here, we are saving the regex in pattern
pattern = re.compile(r"(true|false)")

# Initializing a dictionary data to store the wrangled data
tweet_data = dict()

# Here, we iterate through each item in the created list
# Where, each item is a tweet
for each in list_tweet:

    # Replace our defined words for the text
    # Replacing true with True and false False
    # Using the dummy_dictionary and pattern function
    # This is for finding this particular pattern and substitution
    tweet_words = pattern.sub(lambda x: dummy_dictionary.get(x.group(1), x.group(1)), each)

    # Using eval to get our data into dictionary format
    twitter_eval_dict = eval(tweet_words)

    # Iterate through our evaluated dictionary items
    for key, value in twitter_eval_dict.items():

        # This is as per the specification provided
        if key == 'data':

            # Here, we go through the every item in the list of values
            # Where, every value has the tweet id and the tweet
            for each_item in value:

                # emojis follow utf 16 encoding
                # Here, we handle the utf-16 encoded emojis and bring them to a stand utf-8
                # utf 8 encoding is standard format
                encoded_item = each_item['text'].encode('utf-16','surrogatepass').decode('u

                # Here, we ensure that only the tweets in english are present in the final
                if identifier.classify(encoded_item)[0] == 'en':

                    # Finally, we append the tweet with the created_at date in the format s
                    tweet_data[each_item['id']] = [encoded_item,re.search(r'\d{4}-\d{2}-\d{
```

2.4 Data Extraction

- We now generate a dictionary to sort the tweets according to the dates.
- We then store the wrangled data into an xml file as per the specified format.

In [6]:

```
# Initializing a dictionary date_sorted to store tweets according to dates
date_sorted = {}

# Here, we then change the structure of our wrangled tweet_data
# Here, we group all the tweets according to their created_at date

# Here, we use the sorted function to sort a dictionary by it's value
for key, value in sorted(tweet_data.items()):
    date_sorted.setdefault(value[1], []).append([key,value[0]])

# Final Step :
# Here we are storing and submitting the extracted data into an XML file
# The XML File has been named as per the specification provided

# open: In order to create a new file for writing
# w+ : opens the file for reading and writing
# encoding : utf-8
with open('30945305.xml','w+', encoding='utf-8') as my_xml_file:

    # Adding this data according to the sample xml file
    # write is used
    my_xml_file.write('<?xml version="1.0" encoding="UTF-8"?>\n')

    # We are filling our XML File as per the sample file
    my_xml_file.write('<data>\n')

    # Iterating through the date_sorted dictionary
    for key, values in date_sorted.items():

        my_xml_file.write('<tweets date="%s">\n' % (key))

        for every_item in values:
            my_xml_file.write('<tweet id="%s">%s</tweet>\n' % (every_item[0],every_item[1]))

        my_xml_file.write('</tweets>\n')

    my_xml_file.write('</data>')
```

3. Summary

- Thus, in summary we have successfully performed the following tasks :
  - We have successfully extracted the data in the desired format.
  - We have managed to properly use the langid package.
  - We have ensured the uniqueness of the tweets.
  - We have also managed to create a resultant XML file in the required format.