INFORMATION RETRIEVAL

REAL TIME TWEET LOOKUP

- KARANRAJ M (17PW18)

INFORMATION RETRIEVAL

PHASE-1 REPORT

According to the phase-1 progress, the tweet data is collected from the twitter api (tweepy for python) and necessary preprocessing is done accordingly. The preprocessed data is then indexed and inverting index is completed. There are around 1000 tweets (or documents) taken into consideration for this phase. Out of which, there are around 2207 term dictionaries for which posting lists are created.

WORKING CODE

THE IMPORT STATEMENTS

```
import tweepy as tw

import os

from nltk.tokenize import word_tokenize

from nltk.corpus import stopwords

from dotenv import load_dotenv

load_dotenv(‘.env')
```

THE PREPROCESSING FUNCTIONS

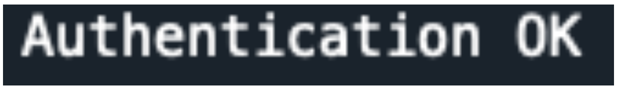
```
def preprocessing(tweet):
    for i in range(len(tweet)):
        temp = tweet[i].split()
        for j in range(len(temp)):
            if temp[j].startswith('https://') or temp[j].startswith('@'):
                temp[j]=""
        tweet[i] = " ".join(temp)
    return tweet

def remove_punctuations(text):
    text=text.lower()
    punc = '''!()-[]{};:'"\,<>./?@#$%^&*_|~'''
    for i in text:
        if i in punc:
            text=text.replace(i,"")
    return text
```

```
def remove_emoji(text):
    char_list = [text[j] for j in range(len(text)) if ord(text[j]) in range(65536)]
    text = ''
    for char in char_list:
        text += char
    return text


def remove_numbers(text):
    temp = text.split()
    for i in range(len(temp)):
        if temp[i].isnumeric():
            temp[i]=""

    text = " ".join(temp)
    return text
```

THE TWEEPY API AUTHENTICATION

```
api_key=os.environ.get('TWITTER_API_KEY')
api_secret_key=os.environ.get('TWITTER_API_SECRET_KEY')
bearer_token=os.environ.get('TWITTER_BEARER_TOKEN')
access_token=os.environ.get('TWITTER_ACCESS_TOKEN')
access_token_secret=os.environ.get('TWITTER_ACCESS_TOKEN_SECRET')
auth = tw.OAuthHandler(api_key, api_secret_key)
auth.set_access_token(access_token, access_token_secret)
api = tw.API(auth, wait_on_rate_limit=True)
try:
    api.verify_credentials()
    print("Authentication OK")
except:
    print("Error during authentication")
```

THE OUTPUT:



```
Authentication OK
```

THE TWEEPY API CALL

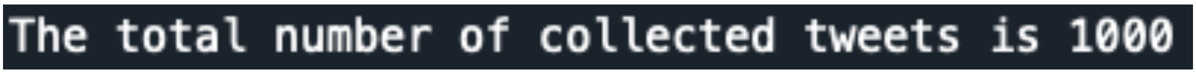
```
search_words = "Olympics"
date_since = "2021-09-21"
api.search
tweets = tw.Cursor(api.search,tweet_mode="extended",
          q=search_words,
          lang="en",
          since=date_since).items(1000)
```

THE DATA COLLECTION

```
texts=[]
mentions=[]
twitterURLs=[]
tweetID=[]
count=0
```

```
  for tweet in tweets:
      texts.append(tweet.full_text)
      mentions.append(tweet.entities['user_mentions'])
      twitterURLs.append(tweet.entities['urls'])
      tweetID.append(tweet.id)
      count+=1

print("The total number of collected tweets is”,count)
```

THE OUTPUT:



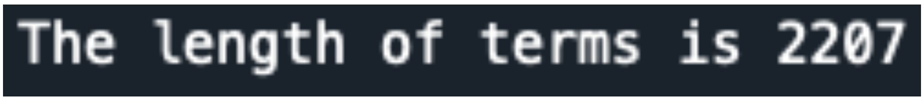
```
The total number of collected tweets is 1000
```

THE PREPROCESSING

```
texts = preprocessing(texts)
```

```
for i in range(len(texts)):
    texts[i] = remove_punctuations(texts[i])
    texts[i] = remove_emoji(texts[i])
    texts[i] = remove_numbers(texts[i])
    text_tokens = word_tokenize(texts[i])
    texts[i] = [word for word in text_tokens if not word in stopwords.words()]


total_tokens=[]
for i in range(len(texts)):
    for j in texts[i]:
        if j not in total_tokens:
            total_tokens.append(j)


print("The length of terms is”,len(total_tokens))
```

THE OUTPUT:



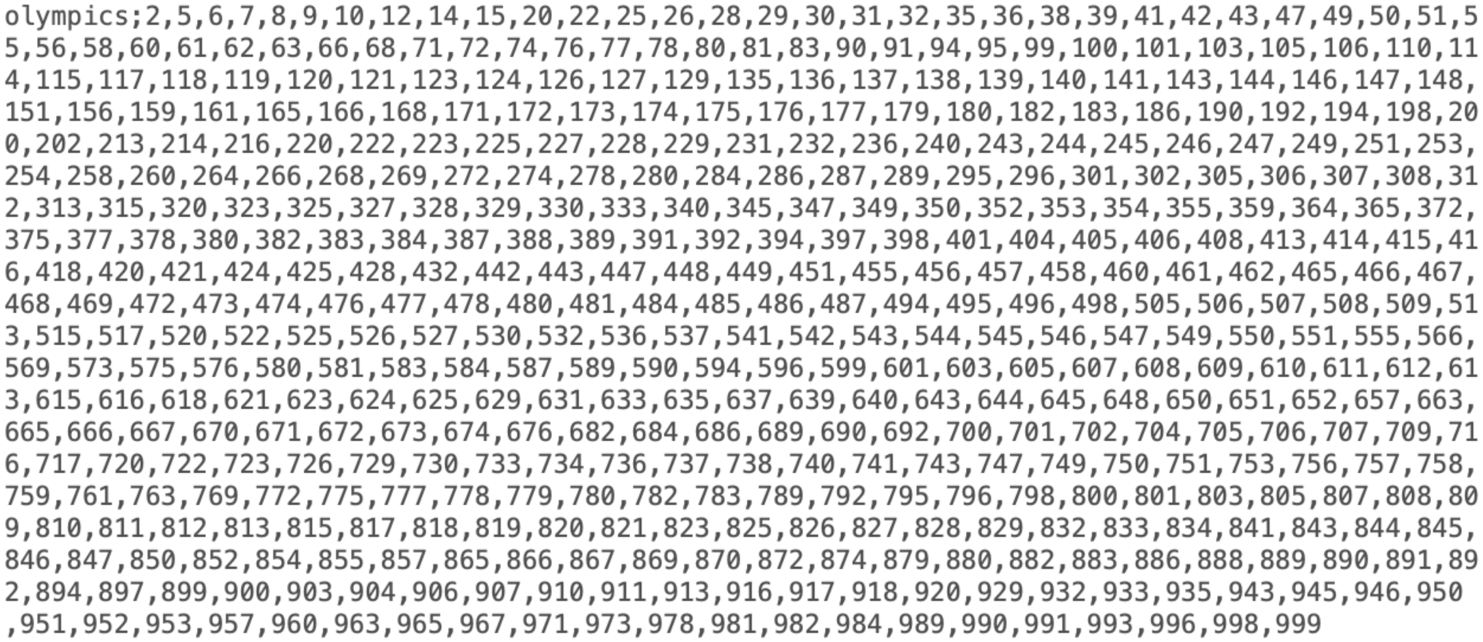
```
The length of terms is 2207
```

THE INDEXING

```
termDict={}
for i in total_tokens:
    temp=[]
    for j in range(len(texts)):
        if i in texts[j]:
            temp.append(j+1)
    termDict[i]=temp

f=open("invertedIndex","w")
for i in termDict:
    termDict[i] = map(str,termDict[i])
    listString = ",".join(termDict[i])
    string = i + ";" + listString + "\n"
    f.write(string)

f.close()
```

THE INVERTED INDEX FILE

The inverted index file contains the inverted indices which are then later used for modeling in the upcoming phases. A small snippet of the file is given below.



```
olympics;2,5,6,7,8,9,10,12,14,15,20,22,25,26,28,29,30,31,32,35,36,38,39,41,42,43,47,49,50,51,5
5,56,58,60,61,62,63,66,68,71,72,74,76,77,78,80,81,83,90,91,94,95,99,100,101,103,105,106,110,11
4,115,117,118,119,120,121,123,124,126,127,129,135,136,137,138,139,140,141,143,144,146,147,148,
151,156,159,161,165,166,168,171,172,173,174,175,176,177,179,180,182,183,186,190,192,194,198,20
0,202,213,214,216,220,222,223,225,227,228,229,231,232,236,240,243,244,245,246,247,249,251,253,
254,258,260,264,266,268,269,272,274,278,280,284,286,287,289,295,296,301,302,305,306,307,308,31
2,313,315,320,323,325,327,328,329,330,333,340,345,347,349,350,352,353,354,355,359,364,365,372,
375,377,378,380,382,383,384,387,388,389,391,392,394,397,398,401,404,405,406,408,413,414,415,41
6,418,420,421,424,425,428,432,442,443,447,448,449,451,455,456,457,458,460,461,462,465,466,467,
468,469,472,473,474,476,477,478,480,481,484,485,486,487,494,495,496,498,505,506,507,508,509,51
3,515,517,520,522,525,526,527,530,532,536,537,541,542,543,544,545,546,547,549,550,551,555,566,
569,573,575,576,580,581,583,584,587,589,590,594,596,599,601,603,605,607,608,609,610,611,612,61
3,615,616,618,621,623,624,625,629,631,633,635,637,639,640,643,644,645,648,650,651,652,657,663,
665,666,667,670,671,672,673,674,676,682,684,686,689,690,692,700,701,702,704,705,706,707,709,71
6,717,720,722,723,726,729,730,733,734,736,737,738,740,741,743,747,749,750,751,753,756,757,758,
759,761,763,769,772,775,777,778,779,780,782,783,789,792,795,796,798,800,801,803,805,807,808,80
9,810,811,812,813,815,817,818,819,820,821,823,825,826,827,828,829,832,833,834,841,843,844,845,
846,847,850,852,854,855,857,865,866,867,869,870,872,874,879,880,882,883,886,888,889,890,891,89
2,894,897,899,900,903,904,906,907,910,911,913,916,917,918,920,929,932,933,935,943,945,946,950
,951,952,953,957,960,963,965,967,971,973,978,981,982,984,989,990,991,993,996,998,999
```

This key indicates the keyword **Olympics** is not present in all the 1000 tweets and therefore they need further modeling to determine the genuinity of those tweets.