Joyce Woznica  
IST 652, Week 9

Activity 9.2: Text Tokenizer

Code:

﻿#----------------------- Question 2 Continued: Read tweets --------------------------

# Import packages for tweets (be sure to start Mongo DB)

import tweepy as tw

import json

import sys

import os

import pymongo

import pandas as pd

import nltk

import nltk

nltk.download('punkt')

nltk.download('stopwords')

# Twitter Keys for API

CONSUMER\_KEY = 'NeMSryEx2Uc6YFi8t74fHsFNe'

CONSUMER\_SECRET = 'XlzCSmkKQM2OhrfO9r10p9fWNRNB1W2Qd8fLOHo0VcCET0ELqM'

OAUTH\_TOKEN = '1259973954166456320-b5xXXSu7WUfTVLEjhXagRvYF6wzmiL'

OAUTH\_SECRET = 'G3AbUXGLUEcobVTeJYlAzz29uiTQIKchSUfb1rVUFgoHI'

auth = tw.OAuthHandler(CONSUMER\_KEY, CONSUMER\_SECRET)

auth.set\_access\_token(OAUTH\_TOKEN, OAUTH\_SECRET)

# db\_fns - courtesy of Dr. D. Landowski

# This function either starts or adds to an existing database and collection in Mongo

# Parameters:

# data - this should be a list of json objects, where each will be a collection element

# in the DB stored under a unique ID key created by Mongo

# DBname - the name of the database, either new or existing

# DBcollection - the name of the collection, either new or existing

def save\_to\_DB (DBname, DBcollection, data):

# connect to database server and just let connection errors fail the program

client = pymongo.MongoClient('localhost', 27017)

# save the results in a database collection

# change names to lowercase because they are not case sensitive

# and remove special characters like hashtags and spaces (other special characters may also be forbidden)

DBname = DBname.lower()

DBname = DBname.replace('#', '')

DBname = DBname.replace(' ', '')

DBcollection = DBcollection.lower()

DBcollection = DBcollection.replace('#', '')

DBcollection = DBcollection.replace(' ', '')

# use the DBname and collection, which will create if not existing

db = client[DBname]

collection = db[DBcollection]

# add the data to the database

collection.insert\_many(data)

print("Saved", len(data), "documents to DB collection", DBname, DBcollection)

# This function gets data from an existing DB and collection

# Parameters:

# DBname and DBcollection- the name of the database and collection, either new or existing

# Result:

# data - returns all the data in the collection as a list of JSON objects

def load\_from\_DB (DBname, DBcollection):

# connect to database server and just let connection errors fail the program

client = pymongo.MongoClient('localhost', 27017)

# use the DBname and collection, which will create if not existing

db = client[DBname]

collection = db[DBcollection]

# get all the data from the collection as a cursor

docs = collection.find()

# convert the cursor to a list

docs\_bson = list(docs)

docs\_json\_str = [dumps(doc) for doc in docs\_bson]

docs\_json = [json.loads(doc) for doc in docs\_json\_str]

return docs\_json

# Simple search using tweepy

# result\_tweets = simple\_search(api, query, max\_results=num\_tweets)

def simple\_search(api, query, since, until, max\_results=500):

# the first search initializes a cursor, stored in the metadata results,

# that allows next searches to return additional tweets

#search\_results = [status for status in tw.Cursor(api.search, q=query).items(max\_results)]

search\_results = tw.Cursor(api.search,

q=search\_words,

lang="en",

since=date\_since,

until=date\_until).items(max\_results)

# for each tweet, get the json representation

tweets = [tweet.\_json for tweet in search\_results]

return tweets

# tie all these together for a single call to all functions

def run\_simple\_tweet\_search(query, num\_tweets, since, until, DBname, DBcollection):

# api = oauth\_login()

''' if needed switch to using the appauth login to avoid rate limiting '''

#api = appauth\_login()

api = tw.API(auth, wait\_on\_rate\_limit=True)

print ("Twitter Authorization: ", api)

# access Twitter search

result\_tweets = simple\_search(api, query, since, until, max\_results=num\_tweets)

tot\_results = len(result\_tweets)

print ('Number of result tweets: ', len(result\_tweets))

# save the results in a database collection

# change names to lowercase because they are not case sensitive

# and remove special characters like hashtags and spaces (other special characters may also be forbidden)

DBname = DBname.lower()

DBname = DBname.replace('#', '')

DBname = DBname.replace(' ', '')

DBcollection = DBcollection.lower()

DBcollection = DBcollection.replace('#', '')

DBcollection = DBcollection.replace(' ', '')

# use the save and load functions in this program

# in this, we do not want any retweeted tweets - only originals

if tot\_results > 0:

save\_to\_DB(DBname, DBcollection, result\_tweets)

# Done!

#-------------------------- Run Tweets -----------------------------

# Set common variables for runs

dbname = "week9db"

collname = "week9coll"

# Final Attempt

import time

import datetime

date\_since = "2019-01-01"

date\_until = time.strftime("%Y-%m-%d")

tweets\_to\_return = 750

search\_wordsList = ["@racingwrongs"]

for search\_words in search\_wordsList:

print(search\_words)

run\_simple\_tweet\_search(search\_words, tweets\_to\_return, date\_since, date\_until, dbname, collname)

# Get Tweet\_List for building pandas dataframe - build one for each tweet\_list

# for racedb

client = pymongo.MongoClient('localhost', 27017)

db = client.week9db

coll = db.week9coll

# Get docs (tweets)

docs = coll.find()

tweets = coll.find()

# Create list of tweets - tweet\_list

doclist = list(docs)

# Create list of tweets - tweet\_list

tweet\_list = [tweet for tweet in tweets]

tweetlist = [doc['text'] for doc in doclist if 'text' in doc.keys()]

len(tweetlist)

all\_tokens = [tok for text in tweetlist for tok in nltk.word\_tokenize(text)]

len(all\_tokens)

# now get how many of each occurs

textFD = nltk.FreqDist(all\_tokens)

textFD.most\_common(30)

# lower case

all\_tokens = [tok.lower() for tweet in tweetlist for tok in nltk.word\_tokenize(tweet)]

# set stopwords

nltk\_stopwords = nltk.corpus.stopwords.words('english')

len(nltk\_stopwords)

import re

def alpha\_filter(w):

pattern = re.compile('^[^a-z]+$')

if (pattern.match(w)):

return True

else:

return False

token\_list = [tok for tok in all\_tokens if not alpha\_filter(tok)]

textFD = nltk.FreqDist(all\_tokens)

top\_words = textFD.most\_common(30)

for word, freq in top\_words:

print(word, freq)

Output:

﻿ @racingwrongs

Twitter Authorization: <tweepy.api.API object at 0x1a24235b90>

Number of result tweets: 105

Saved 105 documents to DB collection week9db week9coll

@ 366

: 230

to 178

the 170

rt 152

racingwrongs 146

. 142

, 136

in 104

at 96

racing 90

of 86

nyclass 82

on 64

https 62

+ 60

may 48

2019 48

pimlico 48

i 42

reopen 40

hold 38

virtual 38

protest 38

response 38

reckless 38

decision 38

horse 36

this 32

; 32