Step 1: Open command prompt with administrator privileg

step 2: cd "C:\Python34\Scripts"

Step 3: C:\Python34\Scripts> pip install --upgrade setuptools

Step 4: pip install nltk

Step 5: Run the program if error is coming then

Step 6: Open Console of python (cmd) and print following command:

>>> import nltk

>>> nltk.download ('stopwords')

>>> nltk.download ('punkt')

—------------------------------------------------------------------------------------------------------------

**CRAWLER**

import requests

from bs4 import BeautifulSoup

def web(page,WebUrl):

if (page>0):

print("Hi!!!!!!!!")

url=WebUrl

code=requests.get(url)

plain=code.text

s=BeautifulSoup(plain,"html.parser")

for link in s.findAll('a'):

tet\_2=link.get('href')

print(tet\_2)

web(1,'https://www.shein.in/promodiscount.html?url\_from=ingoogle brandshein\_shein05\_20190122&gclid=EAIaIQobChMIpteLgIyz4AIVECQrCh29GwEpEAAYASAAEgLrMfD\_BwE')

—--------------------------------------------------------------------------------------------

**STOPWORDS**

from nltk.tokenize import sent\_tokenize,word\_tokenize

from nltk.corpus import stopwords

import nltk

data="This is a sample sentence, showing off the stop words filteration"

print("My text is....")

print(data)

stopwords1=set(stopwords.words('English'))

print("list of stopwords are.....")

print(stopwords1)

words=word\_tokenize(data)

wordsfiltered=[]

for w in words:

if w not in stopwords1:

wordsfiltered.append(w)

print("List of filtered words which are not stopwords \n")

print(wordsfiltered)

—------------------------------------------------------------------------------------------------------------

**XML PARSING**

import csv

import requests

import xml.etree.ElementTree as ET

def loadRSS():

url = 'https://www.w3schools.com/xml/simple.xml'

resp = requests.get(url)

with open('topnewsfeed.xml', 'wb') as f:

f.write(resp.content)

def parseXML(xmlfile):

tree = ET.parse(xmlfile)

root = tree.getroot()

newsitems = []

for item in root.findall('./food'):

news = {}

for child in item:

news[child.tag] = child.text.encode('utf8')

newsitems.append(news)

return newsitems

def savetoCSV(newsitems, filename):

fields = ['name','price', 'description', 'calories']

with open(filename, 'w') as csvfile:

writer = csv.DictWriter(csvfile, fieldnames = fields)

writer.writeheader()

writer.writerows(newsitems)

def main():

loadRSS()

newsitems = parseXML('topnewsfeed.xml')

print(newsitems)

savetoCSV(newsitems, 'topnews.csv')

if \_\_name\_\_ == "\_\_main\_\_":

main()

—-----------------------------------------------------------------------------------------------------------**TWITTER**

import tweepy

from tkinter import \*

from time import sleep

from datetime import datetime

from textblob import TextBlob

import matplotlib.pyplot as plt

consumer\_key = 'AU8Ym4iZoM3deqTn3i6AWDRme'

consumer\_secret = 'h1aT9QPHdtWvjLbPx57di9hnspYFnFT34eG6kTylbC5Tv2C0Nm'

access\_token = '700594962556014592-9NIgRPd60o4YySxlrgQda4uzO97iozW'

access\_token\_secret = 'rPaSPPgEWQZpVS0zVjePj8dhlvkBmvbHZQ0FvswEYygXh'

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

auth.set\_access\_token(access\_token, access\_token\_secret)

api = tweepy.API(auth)

user= api.verify\_credentials()

root = Tk()

label1 = Label(root, text="Search")

E1 = Entry(root, bd =5)

label2 = Label(root, text="Sample Size")

E2 = Entry(root, bd =5)

def getE1():

return E1.get()

def getE2():

return E2.get()

def getData():

getE1()

keyword = getE1()

getE2()

numberOfTweets = getE2()

numberOfTweets = int(numberOfTweets)

polarity\_list = []

numbers\_list = []

number = 1

positive=0

negative=0

neutral=0

for tweet in tweepy.Cursor(api.search\_tweets, keyword,lang="en").items(numberOfTweets):

try:

analysis = TextBlob(tweet.text)

analysis = analysis.sentiment

polarity = analysis.polarity

if polarity>0:

positive+=1

elif polarity <0:

negative+=1

else:

neutral+=1

polarity\_list.append(polarity)

numbers\_list.append(number)

number = number + 1

except tweepy.TweepError as e:

print(e.reason)

except StopIteration:

break

print(polarity)

print(numbers\_list)

print(polarity\_list)

print(f'Amount of positive tweets : {positive}')

print(f'Amount of negative tweets : {negative}')

print(f'Amount of neutral tweets : {neutral}')

axes = plt.gca()

axes.set\_ylim([-1, 1])

plt.scatter(numbers\_list, polarity\_list)

averagePolarity = (sum(polarity\_list))/(len(polarity\_list))

averagePolarity = "{0:.0f}%".format(averagePolarity \* 100)

time = datetime.now().strftime("At: %H:%M\nOn: %m-%d-%y")

plt.text(1, 0.92, "Average Sentiment: " + str(averagePolarity) + "\n" + time, fontsize=12,

bbox = dict(facecolor='none', edgecolor='black', boxstyle='square, pad = 1'))

plt.title("Sentiment of " + keyword + " on Twitter")

plt.xlabel("Number of Tweets")

plt.ylabel("Sentiment")

plt.show()

submit = Button(root, text ="Submit", command = getData)

label1.pack()

E1.pack()

label2.pack()

E2.pack()

submit.pack(side =BOTTOM)

root.mainloop()

—-----------------------------------------------------------------------------------------------------------

**PAGERANK**

def pagerank():

print('Enter the matrix')

array\_input = []

for x in range(3):

array\_input.append([float(y) for y in input().split()])

print(array\_input)

finalmat=[1,1,1]

itr=int(input('Enter the number of iteraions'))

for loop in range(itr):

print('Iteration :', loop+1)

cnt=0

for row in range(len(array\_input)):

sum=0

for col in range(len(array\_input[row])):

if (array\_input[col][row] == 1):

for i in range(3):

if(array\_input[col][i]==1):

cnt=cnt+1

sum+=finalmat[col]/cnt

cnt=0

finalmat[row]=0.5+(0.5\*sum)

print(finalmat[row],' ')

pagerank()

**—------------------------------------------------------------------------------------------------------------**

**INCIDENCE MATRIX**

import pandas as pd

from sklearn.feature\_extraction.text import CountVectorizer

docs=['why hello there','omg hello pony','she went there? omg']

v=CountVectorizer() #count all docs

x=v.fit\_transform(docs) #transform in columnformat

df=pd.DataFrame(x.toarray(),columns=v.get\_feature\_names())

print(df)

—--------------------------------------------------------------------------------------------------------

**COMPARE FILES**

from nltk.tokenize import word\_tokenize

import numpy as np

import nltk

def process(file):

raw=open (file).read()

tokens=word\_tokenize(raw)

count=nltk.defaultdict(int)

for word in tokens:

count[word]+=1

print('file: ', file , count)

return count;

def cos\_sim(a,b):

dot\_product=np.dot(a,b)

norm\_a=np.linalg.norm(a)

print('norm\_a: ',norm\_a)

norm\_b=np.linalg.norm(b)

print('norm\_b: ',norm\_b)

return dot\_product/(norm\_a \* norm\_b)

def getSimilarity(dict1,dict2):

all\_words\_list=[]

for key in dict1:

all\_words\_list.append(key)

for key in dict2:

all\_words\_list.append(key)

print('all\_wors\_list: ',all\_words\_list)

all\_words\_list\_size=len(all\_words\_list)

v1=np.zeros(all\_words\_list\_size,dtype=np.int)

v2=np.zeros(all\_words\_list\_size,dtype=np.int)

i=0

for (key) in all\_words\_list:

v1[i]=dict1.get(key,0)

print('v1',v1)

v2[i]=dict2.get(key,0)

print('v2',v2)

i=i+1

return cos\_sim(v1,v2)

if \_\_name\_\_ == '\_\_main\_\_':

dict1=process("D:/2019-EvenSem/IR/Practical/t1.txt")

dict2=process("D:/2019-EvenSem/IR/Practical/t2.txt")

print("Similarity between two text documents",getSimilarity(dict1,dict2))

**EDIT DISTANCE**

def iterative\_levenshtein(s, t):

rows = len(s)+1

cols = len(t)+1

dist = [[0 for x in range(cols)] for x in range(rows)]

cnt=0

for i in range(1, rows):

cnt=cnt+1

dist[i][0] = cnt

cnt=0

for i in range(1, cols):

cnt=cnt+1

dist[0][i] = cnt

for row in range(1, rows):

for col in range(1, cols):

if s[row-1] == t[col-1]:

dist[row][col] = dist[row-1][col-1]

else:

dist[row][col] = min(dist[row-1][col] + 1,

dist[row][col-1] + 1,

dist[row-1][col-1] + 1)

for r in range(rows):

print(dist[r])

return dist[row][col]

s1=input("Enter String 1 : ")

s2=input("Enter String 2 : ")

print("Edit distance between ",s1," and ",s2," is ",iterative\_levenshtein(s1, s2))

—--------------------------------------------------------

**BITWISE**

plays={"Anthony and Cleopatra":"Anthony is there, Brutus is Caeser is with Cleopatra mercy worser.",

"Julius Ceaser":"Anthony is there, Brutus is Caeser is but Calpurnia is.",

"The Tempest":"mercy worser","Hamlet":"Caeser and Brutus are present with mercy and worser",

"Othello":"Caeser is present with mercy and worser","Macbeth":"Anthony is there, Caeser, mercy."}

words=["Anthony","Brutus","Caeser","Calpurnia","Cleopatra","mercy","worser"]

vector\_matrix=[[0 for i in range(len(plays))] for j in range(len(words))]

text\_list=list((plays.values()))

for i in range(len(words)):

for j in range(len(text\_list)):

if words[i] in text\_list[j]:

vector\_matrix[i][j]=1

else:

vector\_matrix[i][j]=0

for i in vector\_matrix:

print(i)

#result=[]

string\_list=[]

for vector in vector\_matrix:

mystring = ""

for digit in vector:

mystring += str(digit)

string\_list.append(int(mystring,2))

print(string\_list)

print("The output is ", bin(string\_list[0] & string\_list[1] & string\_list[2] & string\_list[3]).replace("0b",""))

#print("The output is ",bin(string\_list[0]& string\_list[1]& (string\_list[2])).replace("0b",""))