Name: Aman Bhardwaj

16BCE0954

Web Mining

Lab Assignment 1

Question:

Write python code to retrieve the relevant document for the given query using cosine similarity index.

Task #1 – perform universal crawling

Task #2 – store the crawled page (at least 50) in a folder

Task #3 – vectorization for documents and term

Task#4 – get a query and find cosine similarity between the given query and every document

Task #5 – rank the documents based on the similarity and display the url of first 10 documents

Code:

1. To implement a universal Crawler

import requests

from bs4 import BeautifulSoup

def crawler(maxpage):

page=1

while page<=maxpage:

url = "https://en.wikipedia.org/wiki/Main\_Page"

source= requests.get(url)

plain = source.text

soup = BeautifulSoup(plain, "lxml")

for link in soup.find\_all('a'):

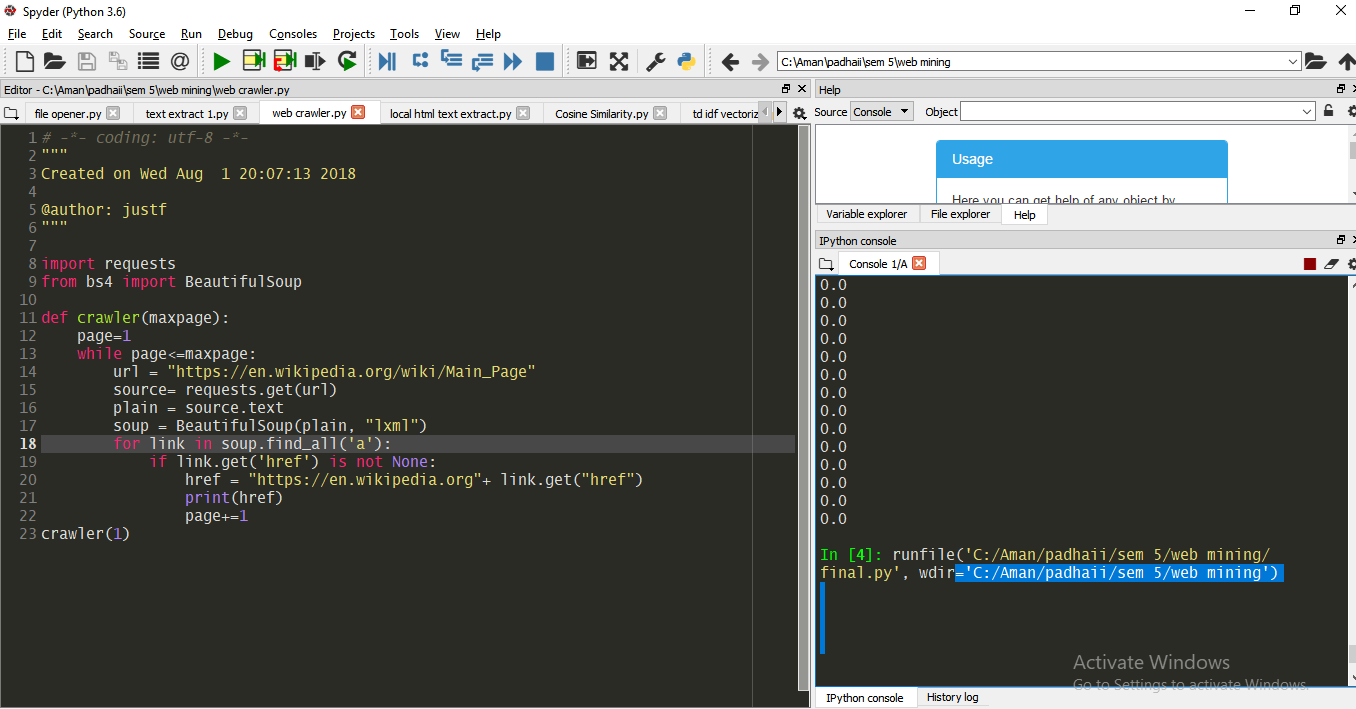
if link.get('href') is not None:

href = "https://en.wikipedia.org"+ link.get("href")

print(href)

page+=1

crawler(1)



b)To save webpages from links extracted from the crawled webpages:

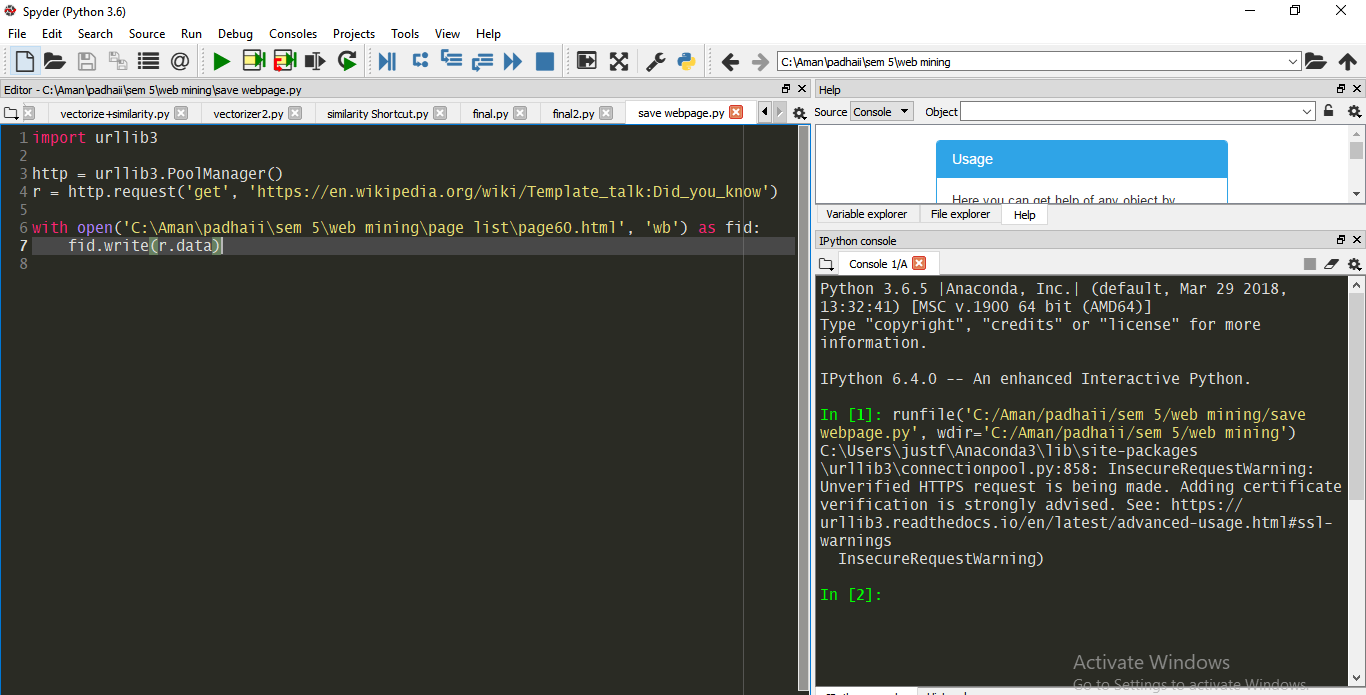
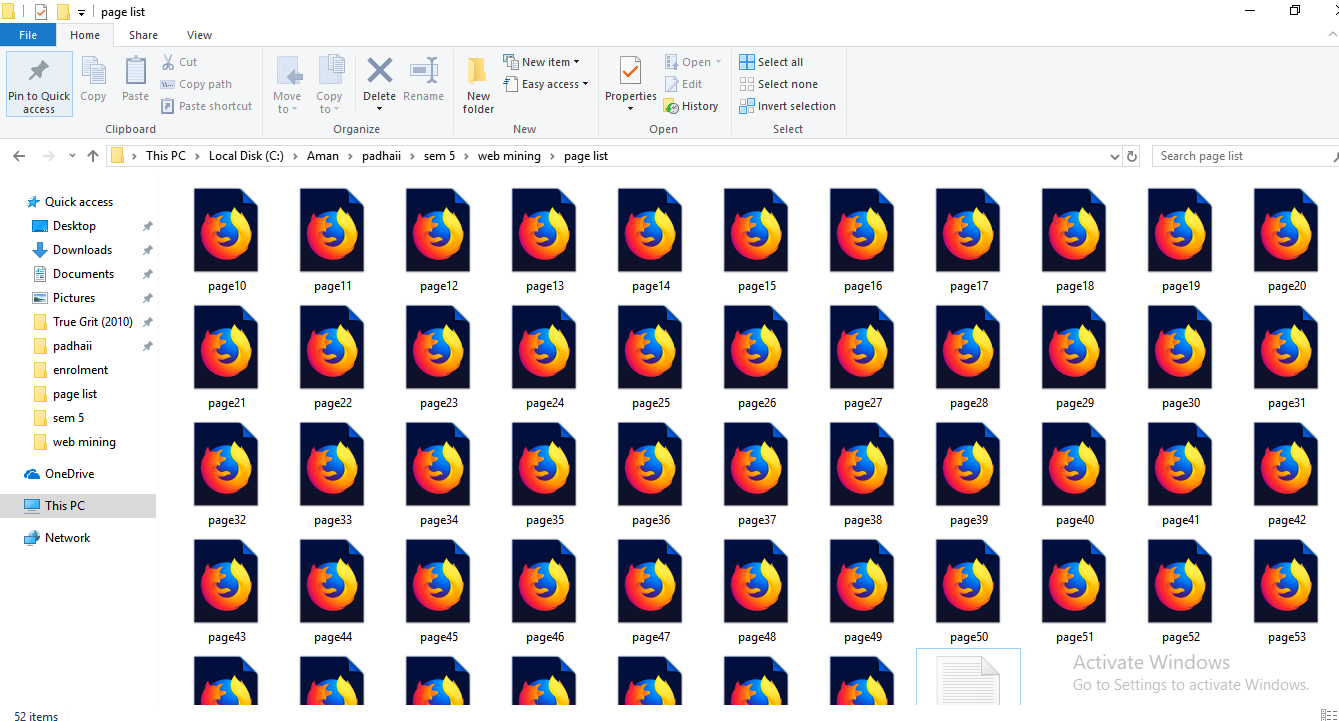
import urllib3

http = urllib3.PoolManager()

r = http.request('get', 'https://en.wikipedia.org/wiki/Template\_talk:Did\_you\_know')

with open('C:\Aman\padhaii\sem 5\web mining\page list\page60.html', 'wb') as fid:

fid.write(r.data)

* Here all the 50 webpages were saved to a local folder
* 
* (The folder containing the saved webpages)
* 

c)To extract the text part from all the saved webpages

from bs4 import BeautifulSoup

from bs4.element import Comment

import urllib.request

def tag\_visible(element):

if element.parent.name in ['style', 'script', 'head', 'title', 'meta', '[document]']:

return False

if isinstance(element, Comment):

return False

return True

def text\_from\_html(body):

soup = BeautifulSoup(body, 'html.parser')

texts = soup.findAll(text=True)

visible\_texts = filter(tag\_visible, texts)

return u" ".join(t.strip() for t in visible\_texts)

file = "C:\Aman\padhaii\sem 5\web mining\page list\page39.html"

f=open(file, 'r+', encoding="utf8")

text = f.read()

mystring = text\_from\_html(text).strip() # the while loop will leave a trailing space,

# so the trailing whitespace must be dealt with

# before or after the while loop

while ' ' in mystring:

mystring = mystring.replace(' ', ' ')

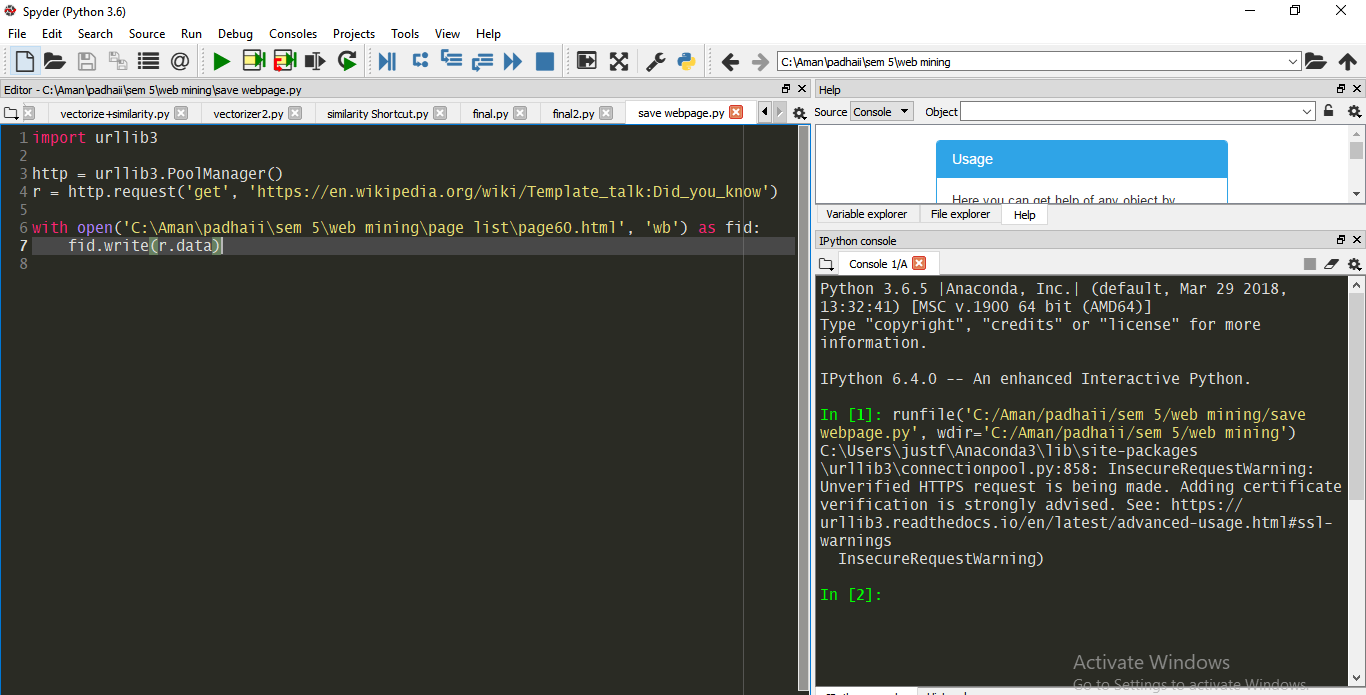
while '...' in mystring:

mystring = mystring.replace('...', '.')

while '^ ' in mystring:

mystring = mystring.replace('^ ', '')

print(mystring)



d)To find cosine similarity between the query(our first saved webpage) , and all the other saved webpages

from bs4 import BeautifulSoup

from bs4.element import Comment

import urllib.request

import re

import math

from collections import Counter

i=11

while i<51:

def tag\_visible(element):

if element.parent.name in ['style', 'script', 'head', 'title', 'meta', '[document]']:

return False

if isinstance(element, Comment):

return False

return True

def text\_from\_html(body):

soup = BeautifulSoup(body, 'html.parser')

texts = soup.findAll(text=True)

visible\_texts = filter(tag\_visible, texts)

return u" ".join(t.strip() for t in visible\_texts)

file = "C:\Aman\padhaii\sem 5\web mining\page list\page10.html"

file1= "C:\Aman\padhaii\sem 5\web mining\page list\page%d.html" % i

f=open(file, 'r+', encoding="utf8")

f1=open(file1, 'r+', encoding="utf8")

text = f.read()

text1 = f1.read()

#print(text\_from\_html(text))

def get\_cosine(vec1, vec2):

intersection = set(vec1.keys()) & set(vec2.keys())

numerator = sum([vec1[x] \* vec2[x] for x in intersection])

sum1 = sum([vec1[x]\*\*2 for x in vec1.keys()])

sum2 = sum([vec2[x]\*\*2 for x in vec2.keys()])

denominator = math.sqrt(sum1) \* math.sqrt(sum2)

if not denominator:

return 0.0

else:

return float(numerator) / denominator

def text\_to\_vector(text):

word = re.compile(r'\w+')

words = word.findall(text)

return Counter(words)

def get\_result(content\_a, content\_b):

text1 = content\_a

text2 = content\_b

vector1 = text\_to\_vector(text1)

vector2 = text\_to\_vector(text2)

cosine\_result = get\_cosine(vector1, vector2)

return cosine\_result

mystring = text\_from\_html(text).strip()

mystring1 = text\_from\_html(text1).strip() # the while loop will leave a trailing space,

# so the trailing whitespace must be dealt with

# before or after the while loop

while ' ' in mystring:

mystring = mystring.replace(' ', ' ')

while ' ' in mystring1:

mystring1 = mystring1.replace(' ', ' ')

while '...' in mystring:

mystring = mystring.replace('...', '.')

while '^ ' in mystring:

mystring = mystring.replace('^ ', '')

while '...' in mystring1:

mystring = mystring1.replace('...', '.')

while '^ ' in mystring1:

mystring = mystring1.replace('^ ', '')

print (get\_result('mystring', 'mystring1'))

i+=1

