B.K. BIRLA COLLEGE OF ARTS, SCIENCE & COMMERCE, KALYAN (Department of Computer Science)

Semester II
Subject Name: Web Mining
Name: Sanjay Jha
Class: M.SC. Computer Science
Roll No.: 7
Exam Seat No.: 2722416

B.K. BIRLA COLLEGE OF ARTS, SCIENCE & COMMERCE, KALYAN

(DEPARTMENT OF COMPUTER SCIENCE)

This is to certify that

Mr./Miss: Sanjay Jha

Roll No.: 07 Exam Seat No.: 2722416

has satisfactorily completed the practical of Web Mining As laid down in the regulation of University of Mumbai for the purpose of

Semester – II Examination:2021 – 2022.

Date: 18 /07 /2022

_____ _____

Professor In-Charge Head of Computer Science Department

Index

Practical 01:

Scrape an online E-Commerce Site for Data.

Code-

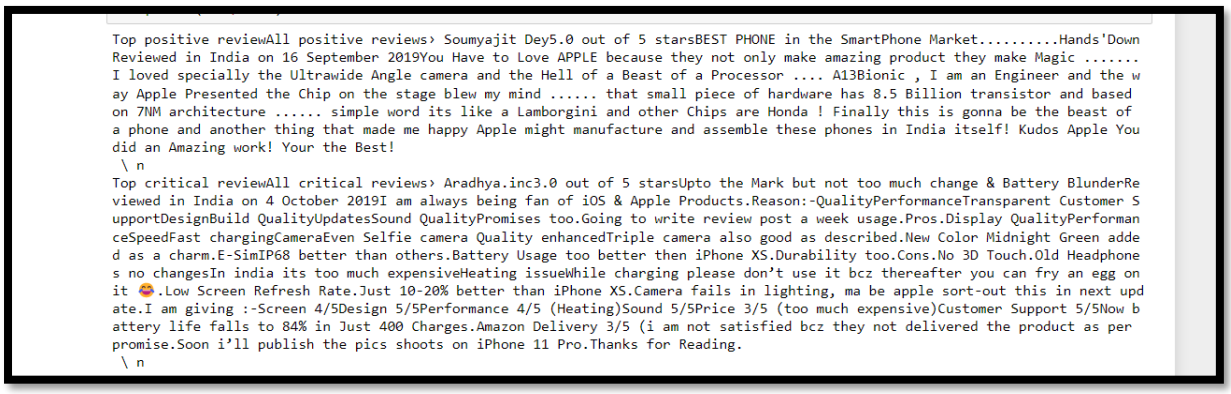
```
pip install beautifulsoup4


# In[ ]:


pip install requests


# In[ ]:


import sys
import time
from bs4 import BeautifulSoup
import requests
import pandas as pd


# In[ ]:


try:
#use the browser to get the url. This is suspicious command that might blow up.
    page=requests.get('https://www.amazon.in/Apple-iPhone-11-Pro-256GB/product-
reviews/B07XVMJF2D/ref=cm_cr_dp_d_show_all_btm?ie=UTF8&reviewerType=all_reviews')

except Exception as e:
   error_type, error_obj, error_info = sys. exc_info()
   print ('ERROR FOR LINK:',url)
   print (error_type, 'Line:', error_info.tb_lineno)
time.sleep(2)
soup=BeautifulSoup (page.text,'html.parser')
links=soup.find_all('div',attrs={'class':'a-expander-content a-expander-partial-collapse-content'})


# In[ ]:


soup


# In[ ]:
```

links

In[ ]:

```
for i in links :
   print ( i.text )
   print ( " \ n " )
```

Output-



```
Top positive reviewAll positive reviews› Soumyajit Dey5.0 out of 5 starsBEST PHONE in the SmartPhone Market.....Hands'Down
Reviewed in India on 16 September 2019You Have to Love APPLE because they not only make amazing product they make Magic .....
I loved specially the Ultrawide Angle camera and the Hell of a Beast of a Processor .... A13Bionic , I am an Engineer and the w
ay Apple Presented the Chip on the stage blew my mind ..... that small piece of hardware has 8.5 Billion transistor and based
on 7NM architecture ..... simple word its like a Lamborgini and other Chips are Honda ! Finally this is gonna be the beast of
a phone and another thing that made me happy Apple might manufacture and assemble these phones in India itself! Kudos Apple You
did an Amazing work! Your the Best!
 \ n
Top critical reviewAll critical reviews› Aradhya.inc3.0 out of 5 starsUpto the Mark but not too much change & Battery BlunderRe
viewed in India on 4 October 2019I am always being fan of iOS & Apple Products.Reason:-QualityPerformanceTransparent Customer S
upportDesignBuild QualityUpdatesSound QualityPromises too.Going to write review post a week usage.Pros.Display QualityPerforman
ceSpeedFast chargingCameraEven Selfie camera Quality enhancedTriple camera also good as described.New Color Midnight Green adde
d as a charm.E-SimIP68 better than others.Battery Usage too better then iPhone XS.Durability too.Cons.No 3D Touch.Old Headphone
s no changesIn india its too much expensiveHeating issueWhile charging please don’t use it bcz thereafter you can fry an egg on
it 😍.Low Screen Refresh Rate.Just 10-20% better than iPhone XS.Camera fails in lighting, ma be apple sort-out this in next upd
ate.I am giving :-Screen 4/5Design 5/5Performance 4/5 (Heating)Sound 5/5Price 3/5 (too much expensive)Customer Support 5/5Now b
attery life falls to 84% in Just 400 Charges.Amazon Delivery 3/5 (i am not satisfied bcz they not delivered the product as per
promise.Soon i’ll publish the pics shoots on iPhone 11 Pro.Thanks for Reading.
 \ n
```

Practical 02:

Perform Spam Classifier

Code-

```
pip install --user -U nltk
```

```
# In[ ]:
```

```
pip install --user -U numpy
```

```
# In[ ]:
```

```
import nltk
nltk.download('punkt')
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from math import log, sqrt
import pandas as pd
import numpy as np
import re
get_ipython().run_line_magic('matplotlib', 'inline')
```

```
# In[ ]:
```

```
mails = pd.read_csv(r"C:\Users\sanjay\Documents\web mining\note\web exam\web p4\spam.csv",
encoding = 'latin-1')
mails.head()
```

```
# In[ ]:
```

```
mails.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis = 1, inplace = True)
mails.head()
```

```
# In[ ]:
```

```
mails.rename(columns = {'v1': 'labels', 'v2': 'message'}, inplace = True)
mails.head()


# In[ ]:


mails['labels'].value_counts()


# In[ ]:


mails['label'] = mails['labels'].map({'ham': 0, 'spam': 1})
mails.head()


# In[ ]:


mails.drop(['labels'], axis = 1, inplace = True)
mails.head()


# In[ ]:


totalMails = 4825 + 747
trainIndex, testIndex = list(), list()
for i in range(mails.shape[0]):
    if np.random.uniform(0, 1) < 0.75:
        trainIndex += [i]
    else:
        testIndex += [i]
trainData = mails.loc[trainIndex]
testData = mails.loc[testIndex]


# In[ ]:


trainData.reset_index(inplace = True)
trainData.drop(['index'], axis = 1, inplace = True)
trainData.head()
```

In[ ]:

```
testData.reset_index(inplace = True)
testData.drop(['index'], axis = 1, inplace = True)
testData.head()
```

In[ ]:

```
trainData['label'].value_counts()
```

In[ ]:

```
testData['label'].value_counts()
```

Output-

Out[12]:

	message	label
0	U dun say so early hor... U c already then say...	0
1	Nah I don't think he goes to usf, he lives aro...	0
2	Even my brother is not like to speak with me. ...	0
3	As per your request 'Melle Melle (Oru Minnamin...	0
4	SIX chances to win CASH! From 100 to 20,000 po...	1

Practical 03:

Demonstrate Text Mining and Webpage Pre-processing using meta information from the web pages (Local/Online).

Code-

```
import nltk
nltk.download('wordnet')


# In[ ]:


import numpy as np
import pandas as pd
import re
import nltk
nltk.download('wordnet')
nltk.download('omw-1.4')
nltk.download('averaged_perceptron_tagger')
import spacy
import string
pd.options.mode.chained_assignment = None

full_df = pd.read_csv(r"C:\Users\sanjay\Documents\web mining\note\web exam\web p3\sample.csv",
nrows=5000)
df = full_df[["text"]]
df["text"] = df["text"].astype(str)
full_df.head()


# In[ ]:


# Lower Casing


# In[ ]:


df["text_lower"] = df["text"].str.lower()
df.head()


# In[ ]:


#Removal of Punctuations


# In[ ]:
```

```
# drop the new column created in last cell
df.drop(["text_lower"], axis=1, inplace=True)

PUNCT_TO_REMOVE = string.punctuation
def remove_punctuation(text):
    """custom function to remove the punctuation"""
    return text.translate(str.maketrans('', '', PUNCT_TO_REMOVE))

df["text_wo_punct"] = df["text"].apply(lambda text: remove_punctuation(text))
df.head()


# In[ ]:


#Removal of Emojis


# In[ ]:


# Reference : https://gist.github.com/slowkow/7a7f61f495e3dbb7e3d767f97bd7304b
def remove_emoji(string):
    emoji_pattern = re.compile("["
                       u"\U0001F600-\U0001F64F"  # emoticons
                       u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                       u"\U0001F680-\U0001F6FF"  # transport & map symbols
                       u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                       u"\U00002702-\U000027B0"
                       u"\U000024C2-\U0001F251"
                       "]+", flags=re.UNICODE)
    return emoji_pattern.sub(r'', string)

remove_emoji("game is on 🔥 🔥 ")


# In[ ]:


remove_emoji("Hilarious😂 ")


# In[ ]:


#Removal of URLs


# In[ ]:
```

```
def remove_urls(text):
    url_pattern = re.compile(r'https?://\S+|www\.\S+')
    return url_pattern.sub(r'', text)


# In[ ]:


text = "Driverless AI NLP blog post on https://www.h2o.ai/blog/detecting-sarcasm-is-difficult-but-ai-may-have-an-answer/"
remove_urls(text)


# In[ ]:


text = "Please refer to link http://lnkd.in/ecnt5yC for the paper"
remove_urls(text)


# In[ ]:


text = "Want to know more. Checkout www.h2o.ai for additional information"
remove_urls(text)


# In[ ]:


#Removal of HTML Tags


# In[ ]:


from bs4 import BeautifulSoup

def remove_html(text):
    return BeautifulSoup(text, "lxml").text

text = """<div>
<h1> H2O</h1>
<p> AutoML</p>
<a href="https://www.h2o.ai/products/h2o-driverless-ai/"> Driverless AI</a>
</div>
"""

print(remove_html(text))


# In[ ]:
```

#Spelling Correction

In[ ]:

pip install pyspellchecker

In[ ]:

```
from spellchecker import SpellChecker

spell = SpellChecker()
def correct_spellings(text):
   corrected_text = []
   misspelled_words = spell.unknown(text.split())
   for word in text.split():
      if word in misspelled_words:
         corrected_text.append(spell.correction(word))
      else:
         corrected_text.append(word)
   return " ".join(corrected_text)

text = "speling correctin"
correct_spellings(text)
```

In[ ]:

```
text = "thnks for readin the notebook"
correct_spellings(text)
```

Output-

	tweet_id	author_id	inbound	created_at	text	response_tweet_id	in_response_to_tweet_id
0	119237	105834	True	Wed Oct 11 06:55:44 +0000 2017	@AppleSupport causing the reply to be disregar...	119236	NaN
1	119238	ChaseSupport	False	Wed Oct 11 13:25:49 +0000 2017	@105835 Your business means a lot to us. Pleas...	NaN	119239.0
2	119239	105835	True	Wed Oct 11 13:00:09 +0000 2017	@76328 I really hope you all change but I'm su...	119238	NaN
3	119240	VirginTrains	False	Tue Oct 10 15:16:08 +0000 2017	@105836 LiveChat is online at the moment - htt...	119241	119242.0
4	119241	105836	True	Tue Oct 10 15:17:21 +0000 2017	@VirginTrains see attached error message. I've...	119243	119240.0

Practical 04:

Scraping Twitter Data using Tweepy library in Python

Code-

```
# Library Imports
import pandas as pd
from bs4 import BeautifulSoup
import requests


# In[ ]:


url = 'https://twitter.com/BillGates?ref_src=twsrc%5Egoogle%7Ctwcamp%5Eserp%7Ctwgr%5Eauthor'

response = requests.get(url)


# In[ ]:


print(response)


# In[ ]:


if response.status_code == 200:
    print(response)


# In[ ]:


# This will store the HTML content as a stream of bytes:
html_content = response.content

# This will store the HTML content as a string:
html_content_string = response.text


# In[ ]:


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


# In[ ]:
```

soup

In[ ]:

appPromoBanner = soup.find('div', {'class':'css-1dbjc4n'})

In[ ]:

appPromoBanner

In[ ]:

all_paragraphs = soup.findAll('p')

In[ ]:

print(all_paragraphs[0:3])

Output-

```
[<p>We’ve detected that JavaScript is disabled in this browser. Please enable JavaScript or switch to a supported browser to co
ntinue using twitter.com. You can see a list of supported browsers in our Help Center.</p>, <p class="errorButton"><a href="htt
ps://help.twitter.com/using-twitter/twitter-supported-browsers">Help Center</a></p>, <p class="errorFooter">
<a href="https://twitter.com/tos">Terms of Service</a>
<a href="https://twitter.com/privacy">Privacy Policy</a>
<a href="https://support.twitter.com/articles/20170514">Cookie Policy</a>
<a href="https://legal.twitter.com/imprint.html">Imprint</a>
<a href="https://business.twitter.com/en/help/troubleshooting/how-twitter-ads-work.html?ref=web-twc-ao-gbl-adsinfo&amp;utm_sour
ce=twc&amp;utm_medium=web&amp;utm_campaign=ao&amp;utm_content=adsinfo">Ads info</a>
      © 2022 Twitter, Inc.
    </p>]
```

Practical 06:

Develop a basic crawler for the web search for user defined keywords.

Code-

```
pip install requests bs4
import logging
from urllib.parse import urljoin
import requests
from bs4 import BeautifulSoup

logging.basicConfig(
   format='%(asctime)s %(levelname)s:%(message)s',
   level=logging.INFO)

class Crawler:

   def __init__(self, urls=[]):
      self.visited_urls = []
      self.urls_to_visit = urls

   def download_url(self, url):
      return requests.get(url).text

   def get_linked_urls(self, url, html):
      soup = BeautifulSoup(html, 'html.parser')
      for link in soup.find_all('a'):
         path = link.get('href')
         if path and path.startswith('/'):
            path = urljoin(url, path)
         yield path

   def add_url_to_visit(self, url):
      if url not in self.visited_urls and url not in self.urls_to_visit:
         self.urls_to_visit.append(url)

   def crawl(self, url):
      html = self.download_url(url)
      for url in self.get_linked_urls(url, html):
         self.add_url_to_visit(url)

   def run(self):
      while self.urls_to_visit:
         url = self.urls_to_visit.pop(0)
         logging.info(f'Crawling: {url}')
         try:
            self.crawl(url)
         except Exception:
            logging.exception(f'Failed to crawl: {url}')
         finally:
            self.visited_urls.append(url)
```

```
if __name__ == '__main__':
    Crawler(urls=['https://www.mcdonalds.com/us/en-us.html']).run()
```

Output-

```
2022-07-10 16:02:26,290 INFO:Crawling: https://www.mcdonalds.com/us/en-us.html
2022-07-10 16:02:26,926 INFO:Crawling: #maincontent
2022-07-10 16:02:26,928 ERROR:Failed to crawl: #maincontent
Traceback (most recent call last):
  File "<ipython-input-1-110dfcb41d28>", line 41, in run
    self.crawl(url)
  File "<ipython-input-1-110dfcb41d28>", line 32, in crawl
    html = self.download_url(url)
  File "<ipython-input-1-110dfcb41d28>", line 17, in download_url
    return requests.get(url).text
  File "C:\Users\sanjay\Anaconda3\lib\site-packages\requests\api.py", line 75, in get
    return request('get', url, params=params, **kwargs)
  File "C:\Users\sanjay\Anaconda3\lib\site-packages\requests\api.py", line 60, in request
    return session.request(method=method, url=url, **kwargs)
  File "C:\Users\sanjay\Anaconda3\lib\site-packages\requests\sessions.py", line 519, in request
    prep = self.prepare_request(req)
  File "C:\Users\sanjay\Anaconda3\lib\site-packages\requests\sessions.py", line 462, in prepare_request
    hooks=merge_hooks(request.hooks, self.hooks),
  File "C:\Users\sanjay\Anaconda3\lib\site-packages\requests\models.py", line 313, in prepare
```