**Web page mining**

**Assignment Specification**

**Description**: This program will extract data from a web page and perform some analysis.

**Input**: No user provided input. Data will be collected from any news website.

**Output**:

Print the headlines

Generate a wordcloud for the words/bigrams in the headlines

Calculate the sentiment

See details in the Procedure.

**Procedure**:

1. Import the needed libraries

These are just some of the libraries I think need to be used

# import the libraries

import bs4 as bs

import requests

from wordcloud import WordCloud

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

1. Define the target URL and open it

# defining the target source

url = 'https://www.dallasnews.com/'

# getting the content

body = requests.get('https://www.dallasnews.com/')

1. Load the page into your “soup” (assuming you are using Beautifulsoup)

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

1. Create an empty list to host the list of words from the headlines
2. Loop into the “soup”, looking for the section with headlines
3. Transform the story heading/headlines into a string first and a list then
4. Remove from the list you created all the non-semantically relevant words (the “stopwords”), using the attached file “stopwords\_en.txt” for the list of stopwords. Feel free to update the list, adding words that may be too frequent and – in your opinion – not too relevant (explaining the reason why you want to remove them). Filter out non-alphabetical elements and perform all the other preliminary cleaning on the text that you may require

#stopwords file is attached

#adding words to the stopwords

stopwords.extend(['dallas', 'texas', 'city'])

1. Looping into the list of clean headlines, print the headlines with the highest and lowest sentiment (3 each)

***This is just an example of how to do the Sentiment Analysis***

##### Sentiment Analysis #####

# calculating the sentiment using vader library

analyzer = SentimentIntensityAnalyzer()

# vader needs strings as input. Transforming the list into string

clean\_text\_str\_pro = ' '.join(Pro\_words)

vad\_sentiment = analyzer.polarity\_scores(clean\_text\_str\_pro)

pos\_pro = vad\_sentiment ['pos']

neg\_pro = vad\_sentiment ['neg']

neu\_pro = vad\_sentiment ['neu']

1. Extract bigram, generating a separate list. Consider bigrams 2 words appearing together more than 2 times in the whole text. Bigrams will be like “word1\_word2”, meaning you will create a new string composed by the 2 words, separated by an underscore (“\_”)
2. Merge the list of single words with the list of bigrams
3. Create a wordcloud with the resulting list. If wordcloud is not available on your computer, either use an online option (see previous assignments) or calculate the sentiment as in previous assignments

***This is an example of how to do the Word Cloud***

##### Word cloud #####

print ('\n\n--- Generating the wordcloud')

# Transforming the lists of words into strings

Pro\_words\_string = ' '.join(Pro\_words)

# Defining the wordcloud parameters

wc = WordCloud(background\_color = "white", max\_words = 2000, stopwords = stopwords)

# Store to file

#wc.to\_file('Pro.png')

# Show the cloud

plt.imshow(wc)

plt.axis('off')

plt.show()

1. Submit the py file