Execute and test your model

Date	06 November 2022
Team ID	PNT2022TMID51318
Project Name	Predicting the energy output of wind turbine based on weather condition

PROGRAM :

```
# Python3 program for a word frequency
# counter after crawling/scraping a web-page
import requests
from bs4 import BeautifulSoup
import operator
from collections import Counter

'''Function defining the web-crawler/core
spider, which will fetch information from
a given website, and push the contents to
the second  function clean_wordlist()'''


def start(url):

    # empty list to store the contents of
    # the website fetched from our web-crawler
    wordlist = []
    source_code = requests.get(url).text

    # BeautifulSoup object which will
    # ping the requested url for data
    soup = BeautifulSoup(source_code, 'html.parser')

    # Text in given web-page is stored under
    # the <div> tags with class <entry-content>
    for each_text in soup.findAll('div', {'class': 'entry-content'}):
        content = each_text.text

        # use split() to break the sentence into
        # words and convert them into lowercase
        words = content.lower().split()

        for each_word in words:
            wordlist.append(each_word)
        clean_wordlist(wordlist)

# Function removes any unwanted symbols
```

```
def clean_wordlist(wordlist):

    clean_list = []
    for word in wordlist:
        symbols = "!@#$%^&*()_-+={[}]|\;:\"<>?/., "

        for i in range(len(symbols)):
            word = word.replace(symbols[i], '')

        if len(word) > 0:
            clean_list.append(word)
    create_dictionary(clean_list)

# Creates a dictionary containing each word's
# count and top_20 occurring words


def create_dictionary(clean_list):
    word_count = {}

    for word in clean_list:
        if word in word_count:
            word_count[word] += 1
        else:
            word_count[word] = 1

    ''' To get the count of each word in
        the crawled page -->

    # operator.itemgetter() takes one
    # parameter either 1(denotes keys)
    # or 0 (denotes corresponding values)

    for key, value in sorted(word_count.items(),
                    key = operator.itemgetter(1)):
        print ("% s : % s " % (key, value))

    <-- '''

    c = Counter(word_count)

    # returns the most occurring elements
    top = c.most_common(10)
    print(top)


# Driver code
if __name__ == '__main__':
    url = "https://www.geeksforgeeks.org/programming-language-choose/"
    # starts crawling and prints output
   start(url)
```