Processing Text Practice Assignment

AUTHOR: 'Darshika Verma'

DATE: '19 June 2024' (Re-submit)

```
pip install spacy
```

```
Requirement already satisfied: spacy in /opt/python/envs/minimal/lib/python3.8/site-packages (3.7.5)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /opt/python/envs/minimal/lib/python3.8/site-packages (from
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from s
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy
Requirement already satisfied: thinc<8.3.0,>=8.2.2 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spa
Requirement already satisfied: weasel<0.5.0,>=0.1.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy)
Requirement already satisfied: typer<1.0.0,>=0.3.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spa
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /opt/python/envs/minimal/lib/python3.8/site-packa
Requirement already satisfied: jinja2 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy) (3.1.3)
Requirement already satisfied: setuptools in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy) (56.0.0)
Requirement already satisfied: packaging>=20.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy) (24.
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spa
Requirement already satisfied: numpy>=1.15.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from spacy) (1.24.
```

```
import spacy
```

```
#!python -m spacy download en_core_web_sm
nlp = spacy.load("en_core_web_sm")
```

```
pip install wordcloud
```

```
Requirement already satisfied: wordcloud in /opt/python/envs/minimal/lib/python3.8/site-packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in /opt/python/envs/minimal/lib/python3.8/site-packages (from wordcloud) (1.
Requirement already satisfied: pillow in /opt/python/envs/minimal/lib/python3.8/site-packages (from wordcloud) (10.2.0)
Requirement already satisfied: matplotlib in /opt/python/envs/minimal/lib/python3.8/site-packages (from wordcloud) (3.7.
Requirement already satisfied: contourpy>=1.0.1 in /opt/python/envs/minimal/lib/python3.8/site-packages (from matplotlib
Requirement already satisfied: cycler>=0.10 in /opt/python/envs/minimal/lib/python3.8/site-packages (from matplotlib->wo
Requirement already satisfied: fonttools>=4.22.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from matplotli
Requirement already satisfied: kiwisolver>=1.0.1 in /opt/python/envs/minimal/lib/python3.8/site-packages (from matplotli
Requirement already satisfied: packaging>=20.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from matplotlib-
Requirement already satisfied: pyparsing>=2.3.1 in /opt/python/envs/minimal/lib/python3.8/site-packages (from matplotlib
Requirement already satisfied: python-dateutil>=2.7 in /opt/python/envs/minimal/lib/python3.8/site-packages (from matplo
Requirement already satisfied: importlib-resources>=3.2.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from
Requirement already satisfied: zipp>=3.1.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from importlib-resou
Requirement already satisfied: six>=1.5 in /opt/python/envs/minimal/lib/python3.8/site-packages (from python-dateutil>=2

[notice] A new release of pip is available: 23.1.2 -> 24.0
[notice] To update, run: pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

```
pip install textacy
```

```
Requirement already satisfied: textacy in /opt/python/envs/minimal/lib/python3.8/site-packages (0.12.0)
Requirement already satisfied: cachetools>=4.0.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy)
Requirement already satisfied: catalogue~=2.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (2.
Requirement already satisfied: cytoolz>=0.10.1 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (0
Requirement already satisfied: jellyfish>=0.8.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (
Requirement already satisfied: joblib>=0.13.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (1.
Requirement already satisfied: networkx>=2.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (3.1
Requirement already satisfied: numpy>=1.17.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (1.2
Requirement already satisfied: pyphen>=0.10.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (0.
Requirement already satisfied: requests>=2.10.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (
Requirement already satisfied: scipy>=0.17.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (1.9
Requirement already satisfied: scikit-learn>=0.19.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textac
Requirement already satisfied: spacy>=3.0.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (3.7.
Requirement already satisfied: tqdm>=4.19.6 in /opt/python/envs/minimal/lib/python3.8/site-packages (from textacy) (4.66
Requirement already satisfied: toolz>=0.8.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from cytoolz>=0.10.
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/python/envs/minimal/lib/python3.8/site-packages (from re
Requirement already satisfied: idna<4,>=2.5 in /opt/python/envs/minimal/lib/python3.8/site-packages (from requests>=2.10
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/python/envs/minimal/lib/python3.8/site-packages (from requests
Requirement already satisfied: certifi>=2017.4.17 in /opt/python/envs/minimal/lib/python3.8/site-packages (from requests
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/python/envs/minimal/lib/python3.8/site-packages (from scikit
```

```
#Load Libraries
from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
from nltk.tokenize import sent_tokenize, word_tokenize
import re
import html
import textacy.preprocessing as tprep
import spacy
# Open Terminal in the Tools Toolbar and type python3 -m spacy download en_core_web_sm
nlp = spacy.load("en_core_web_sm")
```

Step 1: Find a webpage and scrape it

- Find a public html webpage like shown in class and read it into Python using urlopen

The code below performs web scraping on the Wikipedia page about Natural Language Processing. It fetches the content of the page, extracts the main text from the HTML using BeautifulSoup, and then prints the out the characters of the extracted text.

```
links = ['https://en.wikipedia.org/wiki/Natural_language_processing']
url = links[0]  # Extracting the URL from the list
page = urlopen(url).read()
soupified = BeautifulSoup(page, "html.parser")
htmls = soupified.find_all('p')

text = []
for html in htmls:
    text.append(html.get_text().strip())

content = "".join(text)
print(content)
```

```
Natural language processing (NLP) is an interdisciplinary subfield of computer science - specifically Artificial Intelli
Only the introduction of hidden Markov models, applied to part-of-speech tagging, announced the end of the old rule-base
```

Step 2: Fix Errors and Clean text with regex

- Examine the text for errors or problems by looking at the text.
- Use the “impurity” function from class to examine the text for potential issues.
- Remove the noise with the regex function.
- Re-examine the impurity to determine if the data has been mostly cleaned.
- Normalize the rest of the text by using textacy.
- Examine spelling errors in at least one row of the dataset.

Majority of the extracted data was cleaned already since I have extracted only the paragraph part from the wikipedia page. As a result of which, the See Also section was eliminated on it's own and majority of weird characters didn't came.

I utilized the **impurity function** to gauge the proportion of suspicious characters in the initial, uncleaned text. Then, I applied the **clean_up function** to eliminate HTML escapes, tags, URLs, and markdowns from the text. Following this cleaning process, I re-evaluated the impurity of the text using the impurity function. **Notably, the impurity score markedly decreased after the text underwent processing (cleaning) from 0.123 to 0.0**

```
#Use the “impurity” function from class to examine the text for potential issues
import re

# Define the regular expression pattern
RE_SUSPICIOUS = re.compile(r'[&#<>{}\[\]\\]')

def impurity(lst):
    """Returns the share of suspicious characters in a text."""
    if len(lst) < 1:
        return 0
    _sum = 0
    for i in range(len(lst)):
        _sum += len(RE_SUSPICIOUS.findall(lst[i])) / len(lst[i])
    return _sum

# Apply the impurity function to the text
impurity_score = impurity(text)

# Print the result
print("Impurity score:", impurity_score)
```

```
Impurity score: 0.12349053263851772
```

```
#Remove the noise with the regex function
import re
import html

def clean_up(lst):
    # Convert HTML escapes like &amp; to characters.
    for i in range(len(lst)):
        lst[i] = html.unescape(lst[i])
        # Remove HTML tags like <tab>
        lst[i] = re.sub(r'<[^<>]*>', ' ', lst[i])
        # Remove markdown URLs like [Some text](https://....)
        lst[i] = re.sub(r'\[([^\[\]]*)\]\([^\(\)]*\)', r'\1', lst[i])
        # Remove text or code in brackets like [0]
        lst[i] = re.sub(r'\[[^\[\]]*\]', ' ', lst[i])
        # Remove standalone sequences of specials
        lst[i] = re.sub(r'(?:^|\s)[&#<>{}\[\]+|\\:-]{1,}(?:\s|$)', ' ', lst[i])
        # Remove standalone sequences of hyphens like --- or ==
        lst[i] = re.sub(r'(?:^|\s)[\-=\+]{2,}(?:\s|$)', ' ', lst[i])
        lst[i] = re.sub(r'[,\.\(\)-]','',lst[i])
        # Remove sequences of white spaces
        lst[i] = re.sub(r'\s+', ' ', lst[i])
        lst[i] = lst[i].strip()
    return lst

# Apply the clean_up function to the example text
cleaned_text = clean_up(text)

# Print the cleaned text
print("Cleaned text:", cleaned_text)

with open('cleanded_text_chck', 'a') as f:
     for i in cleaned_text:
         f.write(i)
```

```
Cleaned text: ['Natural language processing NLP is an interdisciplinary subfield of computer science specifically Artifi
```

```
#Re-examine the impurity to determine if the data has been mostly cleaned.
# Apply the impurity function to the cleaned text
impurity_score = impurity(cleaned_text)

# Print the result
print("Impurity score:", impurity_score)
```

```
Impurity score: 0.0
```

Step 4: Print output

- Print out the first 1000 characters and the number of tokens in the text.

```
# Print the first 1000 characters of the cleaned text
first_1000_char = ""
char_count = 0
for i in cleaned_text:
    char_count +=len(i)
    if char_count < 1000:
        first_1000_char +=i
print(f'First 1000 characters:',first_1000_char)

#No. of tokens
word_count = 0
for i in cleaned_text:
    word_count +=len(i)
print("\nNumber of tokens:", word_count)
```

```
First 1000 characters: Natural language processing NLP is an interdisciplinary subfield of computer science specifically

Number of tokens: 7224
```

Frequency Table

Before creating the frequency table, I have removed all the stop words since getting the frequencies/no. of ocuurences of 'the', 'and' etc won't make much sense. Although it may alter the meaning of a particular sentence to some extent but now just for the sake of getting the frequency table, I have avoided the stop words.

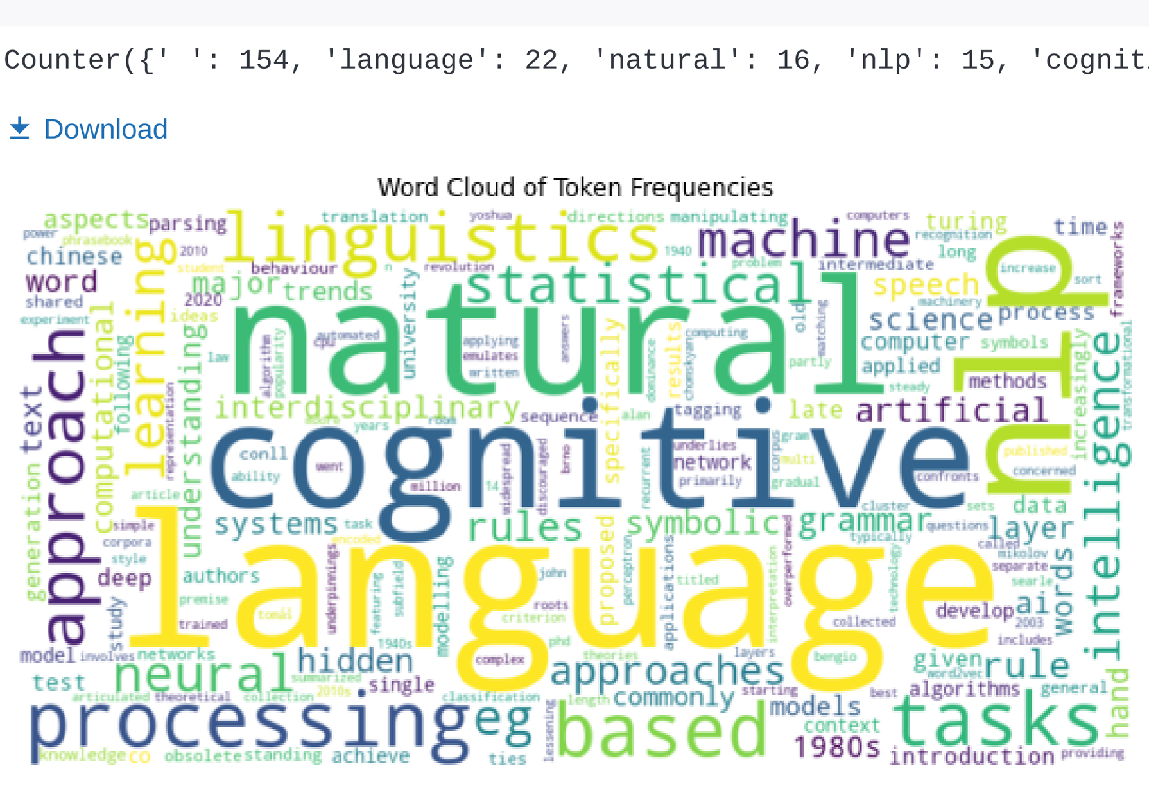
```
# Process the text with spaCy
doc = nlp(cleaned_text_str)
stop_words = spacy.lang.en.stop_words.STOP_WORDS

# Filter out stop words
filtered_text = [token.text for token in doc if token.text.lower() not in stop_words]

# Join the filtered tokens back into a sentence (if needed)
filtered_text_sentence = ' '.join(filtered_text)
```

```
import textacy
from itertools import chain
from collections import Counter
from wordcloud import WordCloud
import matplotlib.pyplot as plt

processed_text = re.sub(r'[,\.\(\)\-\:""]', '', filtered_text_sentence)
doc = textacy.make_spacy_doc(processed_text, lang="en_core_web_sm")
tokenized_lists = [list(token.text.lower() for token in sent) for sent in doc.sents]
frequency_table = Counter(chain.from_iterable(tokenized_lists))
print(frequency_table)

# Generate word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_frequencies(frequency_table)

# Plotting the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Token Frequencies')
plt.show()
```

```
Counter({' ': 154, 'language': 22, 'natural': 16, 'nlp': 15, 'cognitive': 13, 'processing': 11, 'based': 10, 'tasks': 10
```

Download

Word Cloud of Token Frequencies



Interpretation

- Write a paragraph explaining the process of cleaning data for an NLP pipeline. You should explain the errors you found in the dataset and how you fixed them. Discuss the importance of text preprocessing to the NLP pipeline.

Answer

1. Initial inspection of the dataset, such as a scraped Wikipedia page about Natural Language Processing, reveals errors and inconsistencies.

2. The cleaning process streamlines the dataset by eliminating irrelevant content. This ensures that subsequent analyses are focused on pertinent information, enhancing the accuracy and effectiveness of text analysis tasks.

3. After removing irrelevant sections, further processing is performed using specialized functions. For example, the impurity function is utilized to evaluate the presence of suspicious characters in the text.

4. The clean_up function is applied to eliminate HTML escapes, tags, URLs, and markdowns from the text. This enhances the cleanliness and readability of the text, making it more suitable for subsequent analysis.

5. Post-cleaning evaluation reveals a significant decrease in the impurity score of the text. For instance, the impurity score may decrease from 0.123 to 0.00 after the cleaning process.

Importance of Text Preprocessing:

The cleaning and preprocessing steps underscore the importance of text preprocessing in enhancing the quality and reliability of data for NLP pipelines. Meticulous data cleaning ensures that the text is optimized for subsequent analysis, enabling NLP pipelines to yield more accurate insights and results. So, through a systematic cleaning and preprocessing approach, the text becomes more suitable for analysis, thereby enhancing the accuracy and effectiveness of NLP pipelines.

Frequency Table

The frequency table provided insights into the distribution of tokens in the text, revealing common and rare words, as well as potential patterns or trends. From the frequency table, it was observed that certain tokens appeared more frequently than others, indicating their significance or prevalence in the text. The text document, about Natural Language Processing, underwent thorough cleaning and preprocessing, resulting in a refined dataset suitable for subsequent NLP tasks.