Portfolio Assignment 3: Exploring NLTK

Author: Six Wires Instructor: Mazidi Subject: CS 4396 Date: September 10, 2022

1: Imports

```
import nltk


nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('book')
nltk.download('punkt')
nltk.download('omw-1.4')


from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.corpus import wordnet as wn
from nltk.collections import *
from nltk.stem import *
from nltk.book import *
```

3: Use the tokens method to extract 20 tokens from text1

```
print(text1.tokens[:20])
```

```
['[', 'Moby', 'Dick', 'by', 'Herman', 'Melville', '1851', ']', 'ETYMOLOGY', '.',
```

4: Use concordance method on the word "sea"

```
# look for instances of sea and print sentences
print(text1.concordance('sea', 80, 5))
```

```
Displaying 5 of 455 matches:
 shall slay the dragon that is in the sea ." -- ISAIAH " And what thing soever
 S PLUTARCH ' S MORALS . " The Indian Sea breedeth the most and the biggest fis
cely had we proceeded two days on the sea , when about sunrise a great many Wha
many Whales and other monsters of the sea , appeared . Among the former , one w
 waves on all sides , and beating the sea before him into a foam ." -- TOOKE '
None
```

5: Look at the nltk count() method. How is it different from python's count method?

NLTK's count method prints the number of times the parameter word appears in the calling text object, Python's version is case sensitive but seems to do pretty much the same thing.

```
# Create default text
text = "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempo

# Python's count method:
print("Python's count method:", text.count("in"))

# NLTK's count method:
print("NLTK's count method:", text.count("in"))
```

```
Python's count method: 7
NLTK's count method: 7
```

6: Using raw text of at least 5 sentences of your choice from any source (cite the source), save the text into a variable called raw_text. Using NLTK's word tokenizer tokenize the text into variable 'tokens'. Print the first 10 tokens.

Link to text source

```
# Define raw text
raw_text = "He has obstructed the Administration of Justice by refusing his Assent to

# Tokenize the raw text
tokens = word_tokenize(raw_text)

print(tokens[:10])
```

```
['He', 'has', 'obstructed', 'the', 'Administration', 'of', 'Justice', 'by', 'ref
```

7: Using the same raw text, and NLTK's sentence tokenizer sent_tokenize(), perform sentence segmentation and display the sentences.

```
print(sent_tokenize(raw_text))
```

```
['He has obstructed the Administration of Justice by refusing his Assent to Laws
```

8: Using NLTK's PorterStemmer(), write a list comprehension to stem the text. Display the list.

```
# Create stemmer and print out stems
stemmer = PorterStemmer()
stems = [stemmer.stem(word) for word in word_tokenize(raw_text)]

print(stems)
```

```
['he', 'ha', 'obstruct', 'the', 'administr', 'of', 'justic', 'by', 'refus', 'hi'
```

9: Using NLTK's WordNetLemmatizer, write a list comprehension to lemmatize the text Display the list. In the text cell above this code cell, list at least 5 differences you see in the stems verses the lemmas.

Differences:

Stem - Lemma

- he - He
- obstruct - obstructed
- administr - Administration
- justic - Justice

```
# Initialize the lemmatizer
lemmatizer = WordNetLemmatizer()
lemmas = [lemmatizer.lemmatize(word) for word in word_tokenize(raw_text)]

print(lemmas)
```

```
['He', 'ha', 'obstructed', 'the', 'Administration', 'of', 'Justice', 'by', 'refus
```

10: My opinion of the NLTK Library

I think the NLTK library is very in depth and has a meriad of features. The code is very simple to use and easy to understand, and the functions are written cleanly. I could see myself using tokenizers like the one above to analyze sentences and their structure. The lemmatizer seemed to be less harsh than the stemmer, and I may use the lemmatizer to help analyze tone in sentences.