```
import nltk
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('omw-1.4')
nltk.download('book')
from nltk.book import text1
```

```
[nltk_data]    | Downloading package senseval to /root/nltk_data...
[nltk_data]    |   Package senseval is already up-to-date!
[nltk_data]    | Downloading package state_union to /root/nltk_data...
[nltk_data]    |   Package state_union is already up-to-date!
[nltk_data]    | Downloading package stopwords to /root/nltk_data...
[nltk_data]    |   Package stopwords is already up-to-date!
[nltk_data]    | Downloading package swadesh to /root/nltk_data...
[nltk_data]    |   Package swadesh is already up-to-date!
[nltk_data]    | Downloading package timit to /root/nltk_data...
[nltk_data]    |   Package timit is already up-to-date!
[nltk_data]    | Downloading package treebank to /root/nltk_data...
[nltk_data]    |   Package treebank is already up-to-date!
[nltk_data]    | Downloading package toolbox to /root/nltk_data...
[nltk_data]    |   Package toolbox is already up-to-date!
[nltk_data]    | Downloading package udhr to /root/nltk_data...
[nltk_data]    |   Package udhr is already up-to-date!
[nltk_data]    | Downloading package udhr2 to /root/nltk_data...
[nltk_data]    |   Package udhr2 is already up-to-date!
[nltk_data]    | Downloading package unicode_samples to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package unicode_samples is already up-to-date!
[nltk_data]    | Downloading package webtext to /root/nltk_data...
[nltk_data]    |   Package webtext is already up-to-date!
[nltk_data]    | Downloading package wordnet to /root/nltk_data...
[nltk_data]    |   Package wordnet is already up-to-date!
[nltk_data]    | Downloading package wordnet_ic to /root/nltk_data...
[nltk_data]    |   Package wordnet_ic is already up-to-date!
[nltk_data]    | Downloading package words to /root/nltk_data...
[nltk_data]    |   Package words is already up-to-date!
[nltk_data]    | Downloading package maxent_treebank_pos_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package maxent_treebank_pos_tagger is already up-
[nltk_data]    |       to-date!
[nltk_data]    | Downloading package maxent_ne_chunker to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package maxent_ne_chunker is already up-to-date!
[nltk_data]    | Downloading package universal_tagset to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package universal_tagset is already up-to-date!
[nltk_data]    | Downloading package punkt to /root/nltk_data...
[nltk_data]    |   Package punkt is already up-to-date!
[nltk_data]    | Downloading package book_grammars to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package book_grammars is already up-to-date!
[nltk_data]    | Downloading package city_database to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package city_database is already up-to-date!
```

```
[nltk_data]    |   Package city_database is already up-to-date!
[nltk_data]    | Downloading package tagsets to /root/nltk_data...
[nltk_data]    |   Package tagsets is already up-to-date!
[nltk_data]    | Downloading package panlex_swadesh to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package panlex_swadesh is already up-to-date!
[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger is already up-
[nltk_data]    |       to-date!
[nltk_data]    |
[nltk_data]  Done downloading collection book
```

I learned that tokens are a really efficient way to preprocess data. I also learned that the NLTK Text object has a plethora of methods that can be used on text. They all serve a different useful purpose, for instance the corespondance method searches for occurances of strings within the text and the count method finds the number of instances of a particular string in the text.

```
text1.tokens[:20]
```

```
['[',
 'Moby',
 'Dick',
 'by',
 'Herman',
 'Melville',
 '1851',
 ']',
 'ETYMOLOGY',
 '.',
 '(',
 'Supplied',
 'by',
 'a',
 'Late',
 'Consumptive',
 'Usher',
 'to',
 'a',
 'Grammar']
```

```
text1.concordance('sea',lines=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 '
```

This count function essentually finds the number of occurances of a string inside of another string. For instance below text1.count('the') finds the number of instances of 'the' in text1 Both the NLTK Text object count method and this works the same as the regular python count method.

```
text1.count('the')
string = "This is another one of the counting methods using in the Python language. It
string.count('the')
```

```
    13721
```

raw_text from the UTD Wikipedia Page: “University of Texas at Dallas.” Wikipedia, Wikimedia Foundation, 4 Sept. 2022, https://en.wikipedia.org/wiki/University_of_Texas_at_Dallas.

```
raw_text = "The young university has been characterized by rapid growth in research ou

from nltk import word_tokenize
tokens = word_tokenize(raw_text)
print(tokens[:10])
```

```
    ['The', 'young', 'university', 'has', 'been', 'characterized', 'by', 'rapid', 'g
```

```
from nltk import sent_tokenize
sentences = sent_tokenize(raw_text)
print(sentences)
```

```
    ['The young university has been characterized by rapid growth in research output
```

```
from nltk.stem.porter import *

stemmer = PorterStemmer()
stemmed = [stemmer.stem(t) for t in tokens]
print(stemmed)
```

```
    owth', 'in', 'research', 'output', 'and', 'it', 'competit', 'undergradu', 'admiss
```

stems -lemmas 5 differences

univers-university

character-characterized

competit-competitive

undergradu-undergraduate

admiss-admssion

polici-policy

Overall it looks like the lemmas are more accurate than the stems

```
from nltk.stem import WordNetLemmatizer

wnl = WordNetLemmatizer()
lemmatized = [wnl.lemmatize(t) for t in tokens]
print(lemmatized)
```

```
    id', 'growth', 'in', 'research', 'output', 'and', 'it', 'competitive', 'undergrad
```

a. your opinion of the functionality of the NLTK library I belive the functionality of the NLTK library is 2 fold. It serves as a way to learn about the differences between different ways to separate text including lemmas, stems, and tokens and it also functions as a great resource to preprocess text to start a project with text processing.

b. your opinion of the code quality of the NLTK library I think the code quality is very good for the NLTK library, but I believe there is a little to be desired. For instance, the quality of the stemmer and the lemmatizer is not amazing. There are still some stems and lemma which are incorrect and might hinder text processing using the results from them. But it servers as a great way to complete basic text processing functions.

c. a list of ways you may use NLTK in future projects I would use NLTK at the beginning of NLT projects in this class. I think this is a great way to process text before it is analyzed. I would also use it to learn about the differences between stemming and lemmatizing and deciding which is better for a particular project.

✓ 0s completed at 1:51 PM ● ×