Arjun Balasubramanian

axb200075

NLP Portfolio Assignment 2

2. Markdown

In this block we are importing the nltk package as well as downloading some important packages in NLTK that we may use

```
#2
import nltk
nltk.download("stopwords")
nltk.download("wordnet")
nltk.download("punkt")
nltk.download("omw-1.4")
nltk.download("book")
```

```
[nltk_data]    |   Unzipping corpora/ppattach.zip.
[nltk_data]    | Downloading package reuters to /root/nltk_data...
[nltk_data]    | Downloading package senseval to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/senseval.zip.
[nltk_data]    | Downloading package state_union to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/state_union.zip.
[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]    |   Unzipping corpora/swadesh.zip.
[nltk_data]    | Downloading package timit to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/timit.zip.
[nltk_data]    | Downloading package treebank to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/treebank.zip.
[nltk_data]    | Downloading package toolbox to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/toolbox.zip.
[nltk_data]    | Downloading package udhr to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/udhr.zip.
[nltk_data]    | Downloading package udhr2 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/udhr2.zip.
[nltk_data]    | Downloading package unicode_samples to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/unicode_samples.zip.
[nltk_data]    | Downloading package webtext to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/webtext.zip.
[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]    |   Unzipping corpora/wordnet_ic.zip.
```

```
[nltk_data]    | Downloading package words to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/words.zip.
[nltk_data]    | Downloading package maxent_treebank_pos_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping taggers/maxent_treebank_pos_tagger.zip.
[nltk_data]    | Downloading package maxent_ne_chunker to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping chunkers/maxent_ne_chunker.zip.
[nltk_data]    | Downloading package universal_tagset to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping taggers/universal_tagset.zip.
[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]    |   Unzipping grammars/book_grammars.zip.
[nltk_data]    | Downloading package city_database to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/city_database.zip.
[nltk_data]    | Downloading package tagsets to /root/nltk_data...
[nltk_data]    |   Unzipping help/tagsets.zip.
[nltk_data]    | Downloading package panlex_swadesh to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data]    |
[nltk_data]  Done downloading collection book
True
```

3. Markdown

One interesting thing I learned about text objects that I found interesting was that the text in the text objects is stored as tokens rather than as a string, which must be easier for the user as they won't need to go through the process of tokenizing the text themselves. Another interesting thing I learned is that using the text class is much faster than splitting the string as the NLP models can handle it better.

```
#3
from nltk.book import text1 ## This imports the moby dick text from NLTK
text1.tokens[:20] # this gives us the first 20 tokens from the list
```

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

```
 'ETYMOLOGY',
 '.',
 '(',
 'Supplied',
 'by',
 'a',
 'Late',
 'Consumptive',
 'Usher',
 'to',
 'a',
 'Grammar']
```

Double-click (or enter) to edit

```
#4
text1.concordance('sea',lines = 5) ## Here we get 5 concordances for the word 'sea'
```

```
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 '
```

5. Markdown

The count method in the text class is the same as the python count method. The tokens variable is a list that holds all the tokens from the text in it. The count method in text calls the inbuilt python list function count to get the occurrences and returns it back to the user.

```
#5
print("Using the text count method the number of occurences of 'sea' we get is : " + str(text
print("Using the in-built python method the number of occurences of 'sea' we get is : " + str
```

```
Using the text count method the number of occurences of 'sea' we get is : 433
Using the in-built python method the number of occurences of 'sea' we get is : 433
```

MLA citation:

“Shrek Script - Dialogue Transcript.” Shrek Script - Transcript from the Screenplay and/or Mike Myers Movie, http://www.script-o-rama.com/movie_scripts/s/shrek-script-transcript-mike-myers.html.

```
#6
## first 6 lines of shrek
```

```
raw_text = "Once upon a time there was a lovely princess. But she had an enchantment upon her
from nltk.tokenize import word_tokenize ## here we import the tokenizer
tokens = word_tokenize(raw_text) ## Here we tokenize the raw_text and put it in the tokens li
print(tokens[:10]) ## Here we print the tokens
```

```
['Once', 'upon', 'a', 'time', 'there', 'was', 'a', 'lovely', 'princess', '.']
```

```
#7
from nltk.tokenize import sent_tokenize ## import the sentence tokenizer
sent_tokens = sent_tokenize(raw_text) ## create the list using the sent tokenizer
print(sent_tokens) ## printed out the list
```

```
['Once upon a time there was a lovely princess.', "But she had an enchantment upon her c
```

```
#8
from nltk.stem import PorterStemmer ## import the porter stemmer
ps = PorterStemmer() ## create the porter stemmer object
stemmed = [ps.stem(word) for word in tokens] ## using list comprehension to stem the tokensli
print(stemmed) ## print the stemmed words
```

```
['onc', 'upon', 'a', 'time', 'there', 'wa', 'a', 'love', 'princess', '.', 'but', 'she',
```

9. Markdown

onc-once

love-lovely

enchant-enchantment

fear-fearful

onli-only

```
#9
from nltk.stem import WordNetLemmatizer ## import the lemmatizer
lemma = WordNetLemmatizer() ## create the lemmatizer object
lemmad = [lemma.lemmatize(word) for word in tokens] ## use list comprehension to lemmatize th
print(lemmad) ## print the lemmatized words
```

```
['Once', 'upon', 'a', 'time', 'there', 'wa', 'a', 'lovely', 'princess', '.', 'But', 'she
```

10

Overall, I did not have too many issues using the NLTK library. The documentation is well written and I was able to find anything I wanted to do in the documentation. Another useful aspect of NLTK is that is a popular library, so you can find many different other explanations for the code online if you needed a different explanation than on the NLTK website. The quality of the code is also high as the code is open source so there are many optmizations in the code by experts in the NLTK community. In the future there are many applications I could use NLTK for including for neural networks as we can easily break words down into peices for the model, finding similar ideas in text as we can stem to find words with different tenses, or classification to lemmatize words that have different endings.