## **Gutenberg Corpus**

NLTK module includes around 25,000 free books from Gutenberg electronic text archive and it is freely available for us to use.

You can access this corpus from ****nltk.corpus**** module.

from nltk.corpus import gutenberg

If you want to access any text, you’ll have to specify it’s field identifiers.

To check the field identifiers of any text you can use ****fileids()**** function.

>>> gutenberg.fileids()  
['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt', 'bible-kjv.txt', 'blake-poems.txt', 'bryant-stories.txt', 'burgess-busterbrown.txt', 'carroll-alice.txt', 'chesterton-ball.txt', 'chesterton-brown.txt', 'chesterton-thursday.txt', 'edgeworth-parents.txt', 'melville-moby\_dick.txt', 'milton-paradise.txt', 'shakespeare-caesar.txt', 'shakespeare-hamlet.txt', 'shakespeare-macbeth.txt', 'whitman-leaves.txt']

It lists out all the identifiers of the text. If I want to access any text then I can do something like this

>>> gutenberg.words("whitman-leaves.txt")  
['[', 'Leaves', 'of', 'Grass', 'by', 'Walt', 'Whitman', ...]

There are 3 ways to display the content of any text. They are:

* ****Raw****

To display the raw contents of a text, we can use the raw() function. Inside the parenthesis, we should specify the text we are accessing.

>>> gutenberg.raw("whitman-leaves.txt")  
'[Leaves of Grass by Walt Whitman 1855]\n\n\nCome, said my soul,\nSuch verses for my Body let us...........

Raw data will include all the special characters like “\n”, “\t”. These characters are considered as garbage.

* ****Word****

Similar to the raw function, we can use the word() function to display all the words from the context. Words are also considered as tokens.

>>> gutenberg.words("whitman-leaves.txt")  
['[', 'Leaves', 'of', 'Grass', 'by', 'Walt', 'Whitman', ...]

* ****Sentences****

To segregate the text on the basis of sentences, we can use ****sents()**** function.

Where each sentence is a list of words.

>>> gutenberg.sents("whitman-leaves.txt")  
[['[', 'Leaves', 'of', 'Grass', 'by', 'Walt', 'Whitman', '1855', ']'], ['Come', ',', 'said', 'my', 'soul', ',', 'Such', 'verses', 'for', 'my', 'Body', 'let', 'us', 'write', ',', '(', 'for', 'we', 'are', 'one', ',)', 'That', 'should', 'I', 'after', 'return', ',', 'Or', ',', 'long', ',', 'long', 'hence', ',', 'in', 'other', 'spheres', ',', 'There', 'to', 'some', 'group', 'of', 'mates', 'the', 'chants', 'resuming', ',', '(', 'Tallying', 'Earth', "'", 's', 'soil', ',', 'trees', ',', 'winds', ',', 'tumultuous', 'waves', ',)', 'Ever', 'with', 'pleas', "'", 'd', 'smile', 'I', 'may', 'keep', 'on', ',', 'Ever', 'and', 'ever', 'yet', 'the', 'verses', 'owning', '--', 'as', ',', 'first', ',', 'I', 'here', 'and', 'now', 'Signing', 'for', 'Soul', 'and', 'Body', ',', 'set', 'to', 'them', 'my', 'name', ','], ...]

## ****Chat and Web Text****

NLTK module contains different small collections of web text as well as chat text collected over the years.

In the case of web text, it includes content from the firefox discussion forum and some other stuffs.

To access it , you can follow the following steps:

>>> webtext.fileids()  
['firefox.txt', 'grail.txt', 'overheard.txt', 'pirates.txt', 'singles.txt', 'wine.txt']

Naval Postgraduate School has collected text messaging chat sessions for research on detection of internet predators. They have collected chat corpus form, different age groups.

>>> nps\_chat.fileids()['10-19-20s\_706posts.xml', '10-19-30s\_705posts.xml', '10-19-40s\_686posts.xml', '10-19-adults\_706posts.xml', '10-24-40s\_706posts.xml', '10-26-teens\_706posts.xml'.........]

The format of the files are in the format {date-age group-posts}. For example, the file ‘10–19–30s\_705posts.xml’, contains 705 posts gathered from the 30s chat room on 19/10/2006.

## **Corpus in other languages**

NLTK module holds a large collection of data of different languages. For example, if I want to access hindi words from this module then I will have to import “indian” sub module and access language from that.

from nltk.corpus import indianindian.fileids()

## **Creating your own Corpus**

We can easily access our own corpus by specifying the path and loading them with the help of PlaintextCorpusReader.

>>> from nltk.corpus import PlaintextCorpusReader  
>>> directory = "chatbot.txt"  
>>> words = PlaintextCorpusReader(directory,".\*")  
>>> words  
<PlaintextCorpusReader in 'C:\\Users\\ADMIN\\chatbot.txt'>

## ****Conditional Frequency Distribution****

In the previous part, I’ve talked about the Frequency distribution of corpus using ****FreqDist()**** function. To recap, the Frequency distribution is the count of words present in the context. While on the other hand, Conditional Frequency Distribution is the count of words under different conditions.

By conditions, I mean the different categories of text. These conditions could be different genres or on some other basis.

The function used for this purpose is ****ConditionalFreqDist().****This function takes a list of pair where the format is [“condition”, ”word”].

I will use the brown submodule in this case to show an example.

>>>from nltk.corpus import brown  
>>>conditional = ConditionalFreqDist(  
 (genre,words)  
 for genre in ["science\_fiction",'religion']  
 for word in brown.words(categories=genre)]  
>>>conditional  
<ConditionalFreqDist with 2 conditions>

Similar to FreqDist() function we can use the most\_common() function to print out the most common words

>>>onditional["religion"].most\_common()

## ****Stop Words****

There are certain words which are omitted or filtered out before processing documents. These words are known as stop words. Some of the stop words are “a”, “to”, “is”.

NLTK module consists of all the stop words for different languages. We just have to specify it in the parenthesis.

from nltk.corpus import stopwords  
stopwords.words("english")  
>>> from nltk.corpus import stopwords  
>>> stopwords.words("english")  
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself'.......]

# **WordNet**

WordNet is a large collection for lexical database for the English language. I will not be able to explain the whole wordnet module but I will go through something called ****synset.****

Every word in the English language got two or more meanings. They are grouped together and is called a synonym set or synset.

For this example let’s consider the word plane.

>>>from nltk.corpus import wordnet as wn  
>>>wn.synsets("plane")  
[Synset('airplane.n.01'), Synset('plane.n.02'), Synset('plane.n.03'), Synset('plane.n.04'), Synset('plane.n.05'), Synset('plane.v.01'), Synset('plane.v.02'), Synset('plane.v.03'), Synset('flat.s.01')]

The ****synsets()**** function just returns different synonym set of the word “plane”.

>>> wn.synset("airplane.n.01").lemma\_names()  
['airplane', 'aeroplane', 'plane']  
>>> for syn in wn.synsets("plane"):  
... syn.lemma\_names()  
...  
['airplane', 'aeroplane', 'plane']  
['plane', 'sheet']  
['plane']  
['plane', 'planer', 'planing\_machine']  
['plane', "carpenter's\_plane", 'woodworking\_plane']  
['plane', 'shave']  
['plane', 'skim']  
['plane']  
['flat', 'level', 'plane']

In the above code, we can observe that the word “plane” may have different meanings.

Wordnet has many other useful functions and if you want to learn more about it then you should check out the wordnet section of this free book on [nltk](https://www.nltk.org/book/ch02.html" \t "/Users/zcqin/Documents\\x/_blank) [https://www.nltk.org/book/ch02.html](https://www.nltk.org/book/ch02.html" \t "/Users/zcqin/Documents\\x/_blank).