WordNet is a hirarchiacal organization of nouns, verbs, adjectives and adverbs listing short definitions synonym sets use examples relatins to other words

```
import nltk
nltk.download('wordnet')
nltk.download('omw-1.4')
nltk.download('gutenberg')
nltk.download('sentiwornet')
nltk.download('webtext')
nltk.download('nps_chat')
nltk.download('treebank')
nltk.download('inaugural')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data] Downloading package gutenberg to /root/nltk_data...
[nltk_data]   Unzipping corpora/gutenberg.zip.
[nltk_data] Error loading sentiwornet: Package 'sentiwornet' not found
[nltk_data]     in index
[nltk_data] Downloading package webtext to /root/nltk_data...
[nltk_data]   Unzipping corpora/webtext.zip.
[nltk_data] Downloading package nps_chat to /root/nltk_data...
[nltk_data]   Unzipping corpora/nps_chat.zip.
[nltk_data] Downloading package treebank to /root/nltk_data...
[nltk_data]   Unzipping corpora/treebank.zip.
[nltk_data] Downloading package inaugural to /root/nltk_data...
[nltk_data]   Unzipping corpora/inaugural.zip.
True
```

```
from nltk.corpus import wordnet as wn
noun = 'car'
synsets = wn.synsets(noun)
synset = synsets[0]
print(synsets)
print(synset)
```

```
[Synset('car.n.01'), Synset('car.n.02'), Synset('car.n.03'), Synset('car.n.04'), Synset('cable_car.n.01')]
Synset('car.n.01')
```

```
print(synset.definition())
print(synset.examples())
print(synset.lemmas())
```

```
a motor vehicle with four wheels; usually propelled by an internal combustion engine
['he needs a car to get to work']
[Lemma('car.n.01.car'), Lemma('car.n.01.auto'), Lemma('car.n.01.automobile'), Lemma('car.n.01.machine'), Lemma('car.n.01.moto
```

```
hyp = synset.hypernyms()[0]
top = wn.synset('entity.n.01')
while hyp:
  print(hyp)
  if hyp == top:
    break
  if hyp.hypernyms():
    hyp = hyp.hypernyms()[0]
```

```
Synset('motor_vehicle.n.01')
Synset('self-propelled_vehicle.n.01')
Synset('wheeled_vehicle.n.01')
Synset('container.n.01')
Synset('instrumentality.n.03')
Synset('artifact.n.01')
Synset('whole.n.02')
Synset('object.n.01')
Synset('physical_entity.n.01')
Synset('entity.n.01')
```

```
print(synset.hypernyms())
print(synset.hyponyms())
```

```
[Synset('motor_vehicle.n.01')]
[Synset('ambulance.n.01'), Synset('beach_wagon.n.01'), Synset('bus.n.04'), Synset('cab.n.03'), Synset('compact.n.03'), Synset
```

```
print(synset.part_meronyms())
print(synset.part_holonyms())
print(synset.lemmas()[0].antonyms)
```

```
[Synset('accelerator.n.01'), Synset('air_bag.n.01'), Synset('auto_accessory.n.01'), Synset('automobile_engine.n.01'), Synset(
[]
<bound method Lemma.antonyms of Lemma('car.n.01.car')>
```

```
verb = "play"
synsets = wn.synsets(verb)
synset = synsets[0]
print(synsets)

print("Synset chosen is ", synset )
print(synset.definition())
print(synset.examples())
print(synset.lemmas())

hyper = lambda s: s.hypernyms()
list(synset.closure(hyper))
```

```
[Synset('play.n.01'), Synset('play.n.02'), Synset('play.n.03'), Synset('maneuver.n.03'), Synset('play.n.05'), Synset('play.n.
Synset chosen is  Synset('play.n.01')
a dramatic work intended for performance by actors on a stage
['he wrote several plays but only one was produced on Broadway']
[Lemma('play.n.01.play'), Lemma('play.n.01.drama'), Lemma('play.n.01.dramatic_play')]
[Synset('dramatic_composition.n.01'),
 Synset('writing.n.02'),
 Synset('written_communication.n.01'),
 Synset('communication.n.02'),
 Synset('abstraction.n.06'),
 Synset('entity.n.01')]
```

Morphy

```
morph = wn.morphy(verb)
wn.synsets(morph)
```

```
[Synset('play.n.01'),
 Synset('play.n.02'),
 Synset('play.n.03'),
 Synset('maneuver.n.03'),
 Synset('play.n.05'),
 Synset('play.n.06'),
 Synset('bid.n.02'),
 Synset('play.n.08'),
 Synset('playing_period.n.01'),
 Synset('free_rein.n.01'),
 Synset('shimmer.n.01'),
 Synset('fun.n.02'),
 Synset('looseness.n.05'),
 Synset('play.n.14'),
 Synset('turn.n.03'),
 Synset('gambling.n.01'),
 Synset('play.n.17'),
 Synset('play.v.01'),
 Synset('play.v.02'),
 Synset('play.v.03'),
 Synset('act.v.03'),
 Synset('play.v.05'),
 Synset('play.v.06'),
 Synset('play.v.07'),
 Synset('act.v.05'),
 Synset('play.v.09'),
 Synset('play.v.10'),
 Synset('play.v.11'),
 Synset('play.v.12'),
 Synset('play.v.13'),
 Synset('play.v.14'),
 Synset('play.v.15'),
 Synset('play.v.16'),
 Synset('play.v.17'),
 Synset('play.v.18'),
 Synset('toy.v.02'),
 Synset('play.v.20'),
 Synset('dally.v.04'),
 Synset('play.v.22'),
 Synset('dally.v.01'),
 Synset('play.v.24'),
 Synset('act.v.10'),
```

```
 Synset('play.v.26'),
 Synset('bring.v.03'),
 Synset('play.v.28'),
 Synset('play.v.29'),
 Synset('bet.v.02'),
 Synset('play.v.31'),
 Synset('play.v.32'),
 Synset('play.v.33'),
 Synset('meet.v.10'),
 Synset('play.v.35')]
```

```
word_one = "give"
word_two = "provide"

word_one_syn = wn.synsets(word_one)
word_two_syn = wn.synsets(word_two)

print(word_one_syn)
print(word_two_syn)
wup = wn.wup_similarity(word_one_syn[0], word_two_syn[0])
print('Wu_Palmer Similarity: ', wup)
```

```
[Synset('give.n.01'), Synset('give.v.01'), Synset('yield.v.01'), Synset('give.v.03'), Synset('give.v.04'), Synset('give.v.05
[Synset('supply.v.01'), Synset('provide.v.02'), Synset('provide.v.03'), Synset('put_up.v.02'), Synset('leave.v.06'), Synset(
Wu_Palmer Similarity:  0.16666666666666666
```

```
from nltk.wsd import lesk


sent = ['Happy', 'holiday', 'for', 'everyone']
print(lesk(sent, 'happy'))
```

```
Synset('happy.s.04')
```

SentiWordNet is a lexical resource that assigns sentiment scores to synsets in WordNet, a large lexical database of English. It provides a valuable tool for natural language processing tasks such as sentiment analysis, opinion mining, and text classification.

```
import nltk
nltk.download('sentiwordnet')
```

```
[nltk_data] Downloading package sentiwordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/sentiwordnet.zip.
True
```

```
from nltk.corpus import sentiwordnet as swn
word = 'happy'
senti_list = list(swn.senti_synsets(word))
for item in senti_list:
  print(item)
sent = ['Happy', 'holiday', 'for', 'everyone']

for a_word in sent:
  print("For the word: ", a_word)
  senti_list = list(swn.senti_synsets(a_word))
  if senti_list:
    for item in senti_list:
      print(item)
    print()
```

```
<happy.a.01: PosScore=0.875 NegScore=0.0>
<felicitous.s.02: PosScore=0.75 NegScore=0.0>
<glad.s.02: PosScore=0.5 NegScore=0.0>
<happy.s.04: PosScore=0.125 NegScore=0.0>
For the word:  Happy
<happy.a.01: PosScore=0.875 NegScore=0.0>
<felicitous.s.02: PosScore=0.75 NegScore=0.0>
<glad.s.02: PosScore=0.5 NegScore=0.0>
<happy.s.04: PosScore=0.125 NegScore=0.0>

For the word:  holiday
<vacation.n.01: PosScore=0.0 NegScore=0.0>
<holiday.n.02: PosScore=0.0 NegScore=0.0>
<vacation.v.01: PosScore=0.125 NegScore=0.0>

For the word:  for
For the word:  everyone
```

A collocation is a combination of words that are commonly used together in a language or a particular context. These words have a tendency to occur together frequently and their meanings are often difficult to guess from the individual words.

```
import nltk
nltk.download('genesis')
```

```
[nltk_data] Downloading package genesis to /root/nltk_data...
[nltk_data]   Unzipping corpora/genesis.zip.
True
```

```
import nltk
from nltk.book import *
import math
```

```
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
```

```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
text4.collocations()
```

```
United States; fellow citizens; years ago; four years; Federal
Government; General Government; American people; Vice President; God
bless; Chief Justice; one another; fellow Americans; Old World;
Almighty God; Fellow citizens; Chief Magistrate; every citizen; Indian
tribes; public debt; foreign nations
```

```
text = ' '.join(text4.tokens)
vocab = len(set(text4))
hg = text.count('Federal Government')/vocab
print("p(Federal Government) = ",hg )
h = text.count('Federal')/vocab
print("p(Federal) = ", h)
g = text.count('Government')/vocab
print('p(Government) = ', g)
pmi = math.log2(hg/(h*g))
print('pmi = ', pmi)
```

```
p(Federal Government) =  0.0031920199501246885
p(Federal) =  0.006483790523690773
p(Government) =  0.03371571072319202
pmi =  3.868067366919006
```

We see that 'Federal Government' has higher mutual information than indicating it is more likely to be a collocation.

✓ 0s completed at 4:00 PM