



Programación. Python

Librería nltk

nlTK

Introducción

NLTK Downloader

File View Sort Help

Collections Corpora Models All Packages

Identifier	Name	Size	Status
all	All packages	n/a	out of date
all-corpora	All the corpora	n/a	out of date
all-nltk	All packages available on nltk_data gh-pages bran	n/a	out of date
book	Everything used in the NLTK Book	n/a	out of date
popular	Popular packages	n/a	out of date
tests	Packages for running tests	n/a	installed
third-party	Third-party data packages	n/a	installed

Download Refresh

Server Index: https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml

Download Directory: C:\Users\CPAREJA\AppData\Roaming\nltk_data

```
import nltk
nltk.download()
```

showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml

Descarga de textos

*** 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

<Text: Moby Dick by Herman Melville 1851>

```
['[', 'Moby', 'Dick', 'by', 'Herman', 'Melville', '1851', ']', 'ETYMOLOGY', '.', '(', 'Supplied',  
'by', 'a', 'Late', 'Consumptive', 'Usher', 'to', 'a', 'Grammar', 'School', ')', 'The', 'pale', 'Us  
her', '--', 'threadbare', 'in', 'coat', ',', 'heart', ',', 'body', ',', 'and', 'brain', ';', 'I',  
'see', 'him', 'now', '.', 'He', 'was', 'ever', 'dusting', 'his', 'old', 'lexicons', 'and', 'gramma  
rs', ',', 'with', 'a', 'queer', 'handkerchief', ',', 'mockingly', 'embellished', 'with', 'all', 't  
he', 'gay', 'flags', 'of', 'all', 'the', 'known', 'nations', 'of', 'the', 'world', '.', 'He', 'lov  
ed', 'to', 'dust', 'his', 'old', 'grammars', ';', 'it', 'somehow', 'mildly', 'reminded', 'him', 'o  
f', 'his', 'mortality', '.', 'While', 'you', 'take', 'in', 'hand', 'to', 'school', 'others',  
,']
```

Descargamos un libro de ese módulo:

```
from nltk.book import text1 # Moby Dick
```

```
| print(text1)
```

```
print(text1[0:100])
```

Descarga de textos

```
import urllib.request

archivo = urllib.request.urlopen("http://antares.sip.ucm.es/cpareja/")
con_etiquetas = archivo.read()

from bs4 import BeautifulSoup
texto_limpio = BeautifulSoup(con_etiquetas, "lxml")
texto = texto_limpio.get_text(strip=True)

# Veamos un fragmento del texto limpio de etiquetas:

texto[504:887]
```

```
'Áreas e intereses principales:\r\n                                     Programación func  
ional, métodos formales,\r\n                                     transformación de progra  
mas.\r\n                                     Entornos de programación y herramientas de v  
isualización.\r\n                                     Testing and performance evaluation,  
y Cloud computing.\r\n\t\t\t\t\tEnseñanza de la Informática.\r\nHistoria de las matemáticas.'
```

Descarga de textos

```
from urllib import request
```

```
# Descarga de un texto:
```

```
url_1001_noches_part_1 = "http://www.gutenberg.org/cache/epub/47287/pg47287.txt"
```

```
respuesta = request.urlopen(url_1001_noches_part_1)
```

```
mil_y_una_noches = respuesta.read().decode('utf8')
```

```
print(type(mil_y_una_noches))
```

```
print(len(mil_y_una_noches))
```

```
print("-----")
```

```
fragmento_1001_noches = mil_y_una_noches[40150:40582]
```

```
print(fragmento_1001_noches)
```

```
<class 'str'>
```

```
396401
```

```
-----
```

Pero después me acordé de la joya que te destinaba y que te di al llegar á tu palacio. Volví, pues, y encontré á mi mujer acostada con un esclavo negro, durmiendo en los tapices de mi cama. Los maté á los dos, y vine hacia ti, muy atormentado por el recuerdo de tal aventura. Este fué el motivo de mi primera palidez y de mi enflaquecimiento. En cuanto á la causa de haber recobrado mi buen color, dispénsame de mencionarla.»

Tokens

```
from nltk.tokenize import word_tokenize

print(fragmento_1001_noches)

print("-----")

fragmento_1001_noches_tokens = word_tokenize(fragmento_1001_noches)

print(fragmento_1001_noches_tokens)
```

Pero después me acordé de la joya que te destinaba y que te di al llegar á tu palacio. Volví, pues, y encontré á mi mujer acostada con un esclavo negro, durmiendo en los tapices de mi cama. Los maté á los dos, y vine hacia ti, muy atormentado por el recuerdo de tal aventura. Este fué el motivo de mi primera palidez y de mi enflaquecimiento. En cuanto á la causa de haber recobrado mi buen color, dispénsame de mencionarla.»

```
['Pero', 'después', 'me', 'acordé', 'de', 'la', 'joya', 'que', 'te', 'destinaba', 'y', 'que', 't', 'e', 'di', 'al', 'llegar', 'á', 'tu', 'palacio', '.', 'Volví', ',', 'pues', ',', 'y', 'encontré', 'á', 'mi', 'mujer', 'acostada', 'con', 'un', 'esclavo', 'negro', ',', 'durmiendo', 'en', 'los', 'tapices', 'de', 'mi', 'cama', '.', 'Los', 'maté', 'á', 'los', 'dos', ',', 'y', 'vine', 'hacia', 'ti', ',', 'muy', 'atormentado', 'por', 'el', 'recuerdo', 'de', 'tal', 'aventura', '.', 'Este', 'fué', 'el', 'motivo', 'de', 'mi', 'primera', 'palidez', 'y', 'de', 'mi', 'enflaquecimiento', '.', 'En', 'cuanto', 'á', 'la', 'causa', 'de', 'haber', 'recobrado', 'mi', 'buen', 'color', ',', 'dispénsame', 'de', 'mencionarla', '.', '»']
```

Sentences

```
from nltk.tokenize import sent_tokenize

print(fragmento_1001_noches)

print("-----")

fragmento_1001_noches_frases = sent_tokenize(fragmento_1001_noches)

print(fragmento_1001_noches_frases)
```

Pero después me acordé de la joya que te destinaba y que te di al llegar á tu palacio. Volví, pues, y encontré á mi mujer acostada con un esclavo negro, durmiendo en los tapices de mi cama. Los maté á los dos, y vine hacia ti, muy atormentado por el recuerdo de tal aventura. Este fué el motivo de mi primera palidez y de mi enflaquecimiento. En cuanto á la causa de haber recobrado mi buen color, dispénsame de mencionarla.»

```
['Pero después me acordé de la joya que te\r\ndestinaba y que te di al llegar á tu palacio.', 'Volví, pues, y encontré á\r\nmi mujer acostada con un esclavo negro, durmiendo en los tapices de mi\r\ncama.', 'Los maté á los dos, y vine hacia ti, muy atormentado por el\r\nrecuerdo de tal aventura.', 'Este fué el motivo de mi primera palidez y de\r\nmi enflaquecimiento.', 'En cuanto á la causa de haber recobrado mi buen\r\ncolor, dispénsame de mencionarla.»']
```

Stopwords

```
from nltk.corpus import stopwords
```

```
print(type(stopwords))
```

```
stop_espanol = stopwords.words('spanish')
```

```
print(len(stop_espanol))
```

```
print(stop_espanol)
```

```
<class 'nltk.corpus.reader.wordlist.WordListCorpusReader'>
```

```
313
```

```
['de', 'la', 'que', 'el', 'en', 'y', 'a', 'los', 'del', 'se', 'las', 'p  
or', 'un', 'para', 'con', 'no', 'una', 'su', 'al', 'lo', 'como', 'más',  
'pero', 'sus', 'le', 'ya', 'o', 'este', 'sí', 'porque', 'esta', 'entr  
e', 'cuando', 'muy', 'sin', 'sobre', 'también', 'me', 'hasta', 'hay',  
'donde', 'quien', 'desde', 'todo', 'nos', 'durante', 'todos', 'uno', 'l
```

```
... ..
```

```
s', 'tendríamos', 'tendríais', 'tendrían', 'tenía', 'tenías', 'teníamo  
s', 'teníais', 'tenían', 'tuve', 'tuviste', 'tuvo', 'tuvimos', 'tuviste  
is', 'tuvieron', 'tuviera', 'tuvieras', 'tuviéramos', 'tuvierais', 'tuv  
ieran', 'tuviese', 'tuvieses', 'tuviésemos', 'tuvieseis', 'tuviesen',  
'teniendo', 'tenido', 'tenida', 'tenidos', 'tenidas', 'tened']
```


Limpieza: sin stopwords

```
from nltk.corpus import stopwords

def removeStopwords(palabras):
    return [word for word in palabras if word not in stopwords.words('spanish')]

fragmento_2001_sin_stop = removeStopwords(fragmento_1001_noches_tokens)

print(fragmento_2001_sin_stop)
```

Pero después me acordé de la joya que te destinaba y que te di al llegar á tu palacio. Volví, pues, y encontré á mi mujer acostada con un esclavo negro, durmiendo en los tapices de mi cama. Los maté á los dos, y vine hacia tí, muy atormentado por el recuerdo de tal aventura. Este fué el motivo de mi primera palidez y de mi enflaquecimiento. En cuanto á la causa de haber recobrado mi buen color, dispénsame de mencionarla.»

```
['Pero', 'después', 'acordé', 'joya', 'destinaba', 'di', 'llegar', 'á',  
'palacio', '.', 'Volví', ',', 'pues', ',', 'encontré', 'á', 'mujer', 'a  
costada', 'esclavo', 'negro', ',', 'durmiendo', 'tapices', 'cama', '.',  
'Los', 'maté', 'á', 'dos', ',', 'vine', 'hacia', ',', 'atormentado', 'r  
ecuerdo', 'tal', 'aventura', '.', 'Este', 'fué', 'motivo', 'primera',  
'palidez', 'enflaquecimiento', '.', 'En', 'cuanto', 'á', 'causa', 'habe  
r', 'recobrado', 'buen', 'color', ',', 'dispénsame', 'mencionarla',  
'.', '»']
```

Frecuencia de palabras significativas

```
mil_y_una_noches_tokens = word_tokenize(mil_y_una_noches)
mil_y_una_noches_tokens_sin_stop = removeStopwords(mil_y_una_noches_tokens)
frecuencias_terminos_1001_noches = nltk.FreqDist(mil_y_una_noches_tokens_sin_stop)
terms_frecs = list(frecuencias_terminos_1001_noches.items())
terms_frecs.sort(key=lambda par: par[1], reverse=True)

for termino, frec in terms_frecs[:25]:
    print(termino + " -> " + str(frec))
```

```
, -> 5198
. -> 2201
á -> 1446
: -> 1083
Y -> 949
« -> 910
! -> 900
» -> 897
dijo -> 403
```

```
rey -> 321
Entonces -> 281
? -> 235
joven -> 198
Alah -> 196
_ -> 186
Pero -> 181
[ -> 177
] -> 177
```

```
; -> 176
¡Oh -> 164
the -> 162
-- -> 162
El -> 138
¡oh -> 127
efrit -> 125
```

Frecuencia de palabras significativas: gráfico

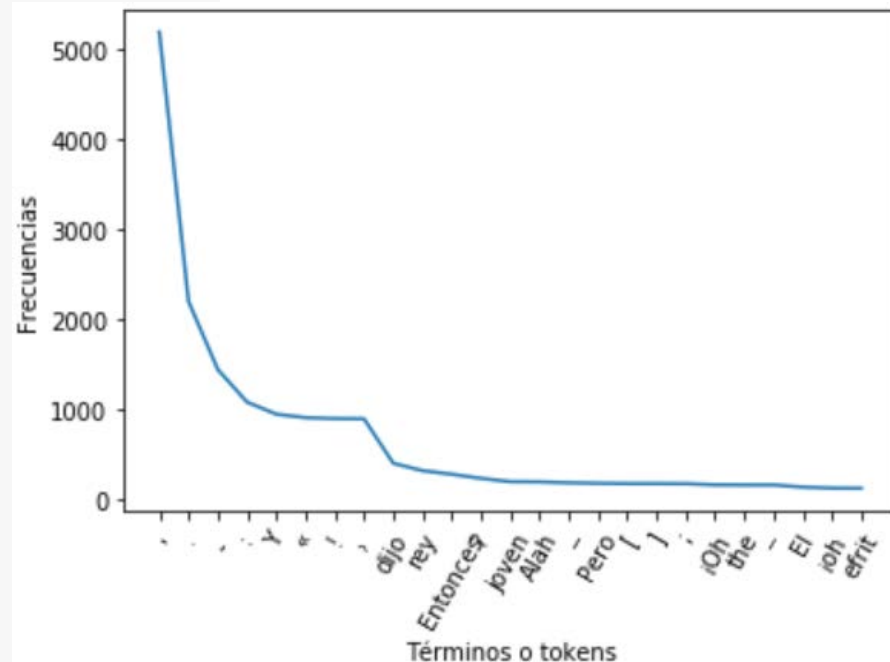
```
import matplotlib.pyplot as plt
import numpy as np

abcisas = [x for (x,_y) in terms_frecs[:25]]
valores = [y for (_, y) in terms_frecs[:25]]

x_pos = np.arange(len(abcisas))

plt.plot(x_pos, valores)
plt.xticks(x_pos, abcisas, rotation=60)

plt.ylabel('Frecuencias')
plt.xlabel('Términos o tokens')
plt.show()
```



Sinónimos Y antónimos

```
from nltk.corpus import wordnet  
  
sinonimos = wordnet.synsets("big")  
  
print(sinonimos)  
  
print(sinonimos[0].definition())  
print(sinonimos[0].examples())
```

```
[Synset('large.a.01'), Synset('big.s.02'), Synset('bad.s.02'), Synset('big.s.04'), Synset('big.s.05'), Synset('big.s.06'), Synset('boastful.s.01'), Synset('big.s.08'), Synset('adult.s.01'), Synset('big.s.10'), Synset('big.s.11'), Synset('big.s.12'), Synset('big.s.13'), Synset('big.r.01'), Synset('boastfully.r.01'), Synset('big.r.03'), Synset('big.r.04')]
```

above average in size or number or quantity or magnitude or extent

```
['a large city', 'set out for the big city', 'a large sum', 'a big (or large) barn', 'a large family', 'big businesses', 'a big expenditure', 'a large number of newspapers', 'a big group of scientists', 'large areas of the world']
```

Sinónimos y antónimos

```
sinonimos, antonimos = [], []
```


```
def sinonimos_de(palabra):  
    sinonimos = [lemma.name() for  
                  sin in wordnet.synsets(palabra)  
                  for lemma in sin.lemmas()]  
    return sinonimos  
  
def antonimos_de(palabra):  
    antonimos = [lemma.antonyms()[0].name()  
                 for sin in wordnet.synsets(palabra)  
                 for lemma in sin.lemmas() if lemma.antonyms()]  
    return antonimos
```

```
print(sinonimos_de("big"))  
print(antonimos_de("big"))
```

```
['large', 'big', 'big', 'bad', 'big', 'big', 'big', 'large', 'prominent',  
'big', 'heavy', 'boastful', 'braggart', 'bragging', 'braggy', 'big', 'cock-  
a-hoop', 'crowing', 'self-aggrandizing', 'self-aggrandising', 'big', 'swell  
ed', 'vainglorious', 'adult', 'big', 'full-grown', 'fully_grown', 'grown',  
'grownup', 'big', 'big', 'large', 'magnanimous', 'big', 'bighearted', 'boun  
teous', 'bountiful', 'freehanded', 'handsome', 'giving', 'liberal', 'openha  
nded', 'big', 'enceinte', 'expectant', 'gravid', 'great', 'large', 'heavy',  
'with_child', 'big', 'boastfully', 'vauntingly', 'big', 'large', 'big', 'bi  
g']  
['small', 'little', 'small']
```


Derivación regresiva o *stemming*

```
from nltk.stem import PorterStemmer  
  
stemmer = PorterStemmer()  
  
print(stemmer.stem('working'))  
print(stemmer.stem('worked'))  
print(stemmer.stem('works'))
```



work
work
work

```
from nltk.stem import SnowballStemmer  
raiz_espannola = SnowballStemmer("spanish")  
  
print(raiz_espannola.stem('trabajaba'))  
print(raiz_espannola.stem('trabajos'))  
print(raiz_espannola.stem('trabajoso'))  
print(raiz_espannola.stem('trabajaré'))  
print(raiz_espannola.stem('trabajar'))  
print(raiz_espannola.stem('trabajando'))  
print(raiz_espannola.stem('trabajaríamos'))
```



trabaj
trabaj
trabaj
trabaj
trabaj
trabaj
trabaj

Referencias

- <https://likegeeks.com/es/tutorial-de-nlp-con-python-nltk/>
- <https://www.nltk.org/book/ch01.html>
- <https://pmoracho.github.io/blog/2017/01/04/NLTK-mi-tutorial/>