TOOL

NLTK Library

Natural Language Toolkit (NLTK) is a library of statistical and symbolic tools, sample corpora, and databases for natural language processing. This certificate relies heavily on the algorithms and sample documents in this package. In this tool, you are introduced to the essential uses of NLTK. You can find a more thorough introduction to the NLTK package in the free **NLTK book**.

Import and Download Corpora

NLTK is imported just like any other library in Python, but there is a catch. The package comes with numerous algorithms, but dozens of corpora are not yet downloaded to conserve gigabytes of your computer's storage space. In any given project or task, you are likely to use only a small number of corpora and databases (containing document files in .txt or other formats).

Typically, users explicitly specify which files need to be downloaded, which is done only once, unless your Python session is reset and the storage is wiped, which happens with many free public computing services.

Use the `download()` method to specify a list of corpora (string) names, each possibly containing multiple documents or databases. The status result is returned into a dummy variable named _ .

```python
import nltk
_ = nltk.download([‘brown’], quiet=True)  # quietly download Brown corpus
```

To view the full list of corpora, execute `nltk.download()`. Then either a downloader will open in a separate window (if Python is running locally via integrated development environment or IDE) or you will see the following menu allowing you to navigate through hundreds of corpora and databases:

```
NLTK Downloader
---------------------------------------------------------------------------
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
Downloader>
```

This download() method is fairly flexible and allows you to specify files in a given directory, if you do not wish to use the occasionally changing default set of files from NLTK developers and contributors.

Loading a Corpus

After a corpus is downloaded, you can load it in your code. As an example, the **Brown Corpus**, or Brown University Standard Corpus of Present-Day American English, is a typical corpus that contains a collection of 500 carefully curated documents (or sources such as news, editorial, etc.) with about 50,000 unique words or one million words in total.

Below is a table describing several first and last documents in the Brown Corpus, ordered by their increasing counts of words. For example, a document *cj06.txt* contains 81 sentences, 2,148 words, and 19,690 characters. The raw column shows the first few words of this document.

	brown	sents	words	char	raw
299	cj06	81	2,148	19,690	The thermal exchange of chlorine between Af and liquid Af is readily measurable at temperatures...
325	cj32	99	2,170	18,725	The many linguistic techniques for reducing the amount of dictionary information that have been...
333	cj40	70	2,171	19,353	In assessing the outlook for interest rates in 1961, the question, as always, is the prospect...
331	cj38	95	2,174	20,249	Unemployed older workers who have no expectation of securing employment in the occupation in whi...
...	...	...	...	...	...
285	ch22	83	2,259	23,775	Purchase authorizations will include provisions relating to the sales and delivery of commodities...
461	cn29	192	2,560	20,278	“Bastards,“ he would say, “all I did was put a beat to that Vivaldi stuff, and the firs

	brown	sents	words	char	raw
84	cc14	102	2,574	22,531	Elisabeth Schwarzkopf sang so magnificently Saturday night at Hunter College that it seems a pit…
493	cr03	92	2,587	21,655	Needless to say, I was furious at this unparalleled intrusion upon free enterprise. How dared…

A typical corpus of documents offers several ways to retrieve text. The **`word() method`** returns a list of words in a corpus.

```
> nltk.corpus.brown.words()  # returns a list of words in Brown corpus
[‘The’, ‘Fulton’, ‘County’, ‘Grand’, ‘Jury’, ‘said’, ...]
```

The **`sents() method`** returns a list of sentences, each of which is a list of words, in a corpus.

```
> nltk.corpus.brown.sents()
[[‘The’, ‘Fulton’, ‘County’, ‘Grand’, ‘Jury’, ‘said’, ‘Friday’, ‘an’,
‘investigation’, ‘of’, “Atlanta’s”, ‘recent’, ‘primary’, ‘election’,
‘produced’, ‘``’, ‘no’, ‘evidence’, “’’”, ‘that’, ‘any’, ‘irregularities’,
‘took’, ‘place’, ‘.’], ...]
```

The **`raw() method`** returns the original unparsed and unprocessed text containing newline characters, tabs, etc. It is great for trying out various preprocessing techniques.

```
> nltk.corpus.brown.raw()
‘\n\n\tThe/at Fulton/np-tl County/nn-tl Grand/jj-tl Jury/nn-tl said/vbd
Friday/nr an/at investigation/nn of/in Atlanta’s/np$ recent/jj primary/nn
election/nn produced/vbd ``/`` no/at evidence/nn …’
```

Learning More NLTK Corpus Methods

Chapter 02 of NLTK book dives deeper into arguments of corpus methods. For example, instead of retrieving all documents from Brown, you can specify a specific document.

```
> from nltk.corpus import brown
> brown.words(fileids=[‘cj06’])
[‘The’, ‘thermal’, ‘exchange’, ‘of’, ‘chlorine’, ...]
```

Also, you can view available methods via the `dir()` command:

```
> ‘, ‘.join([t for t in dir(brown) if ‘_’ not in t]) # show user methods
‘abspath, abspaths, categories, citation, encoding, fileids, license, open,
paras, raw, readme, root, sents, words’
```

Identifying Document Files in a Corpus

If unsure about the contents of a given corpus, you can list all the document files it contains using the `fileids()` method of the corpus object. For example, the `names` corpus contains two documents: a document with a list of common female names and a document with a list of common male names (one name per line).

```
> _ = nltk.download([‘names’], quiet=True)
> nltk.corpus.names.fileids()
[‘female.txt’, ‘male.txt’]
```

NLTK Algorithms

The NLTK package has a number of simple and useful algorithms. For example, `nltk.sent_tokenize()` is a simple algorithm that splits a document into a list of sentences, while `nltk.word_tokenize()` splits a document into a list of words. This certificate will also cover many other specific text splitting and preprocessing algorithms.