Text Mining Online | Text Analysis Online | Text Processing Online

Text Mining | Text Analysis | Text Process | Natural Language Processing

Dive Into NLTK, Part I: Getting Started with NLTK

NLTK is the most famous Python Natural Language Processing Toolkit, here I will give a detail tutorial about NLTK. This is the first article in a series where I will write everything about NLTK with Python, especially about text mining and text analysis online.

This is the first article in the series "Dive Into NLTK", here is an index of all the articles in the series that have been published to date:

About NLTK

Here is a descripiton from the NLTK official site:

> *NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.*

Installing NLTK

The following step is test on my mac os and a vps with ubuntu 12.04, just require your computer with Python 2.6 or Python 2.7, but I did't test it on a windows computer. And I assume you could write some python code, and familiarity with Python modules and packages is also recommended. Here is the step to install NLTK on Mac/Unix:

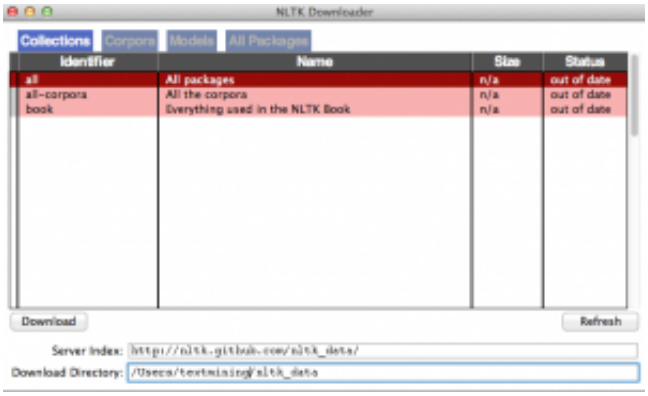
```
Install Setuptools: http://pypi.python.org/pypi/setuptools
Install Pip: run sudo easy_install pip
Install Numpy (optional): run sudo pip install -U numpy
Install PyYAML and NLTK: run sudo pip install -U pyyaml nltk
Test installation: run python then type import nltk
```

Installing NLTK Data

After installing NLTK, you need install NLTK Data which include a lot of corpora, grammars, models and etc. Without NLTK Data, NLTK is nothing. You can find the complete nltk data list here: http://nltk.org/nltk_data/

The simplest way to install NLTK Data is run the Python interpreter and type the commands, following example is running on Mac Os:
[GCC 4.2.1 Compatible Apple Clang 4.0 (tags/Apple/clang-418.0.60)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.download()

A new window should open, showing the NLTK Downloader on Mac(Maybe same on Windows):



Click on the File menu and select Change Download Directory, next, select the packages or collections you want to download, we suggest you select the "all" and download everything NLTK needed.

Graphical interface

If you install NLTK Data in a linux vps, no graphical interface, no window open, you still can use above

nltk.download() command, you can following the follow step to download all nltk_data:

```
Python 2.7.3 (default, Sep 26 2013, 20:03:06)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.download()
NLTK Downloader
---------------------------------------------------------------------------
d) Download l) List u) Update c) Config h) Help q) Quit
---------------------------------------------------------------------------
Download which package (l=list; x=cancel)?
Downloader> l

Packages:
[*] maxent_ne_chunker... ACE Named Entity Chunker (Maximum entropy)
[*] abc................. Australian Broadcasting Commission 2006
[*] alpino.............. Alpino Dutch Treebank
[*] biocreative_ppi..... BioCreAtIvE (Critical Assessment of Information
Extraction Systems in Biology)
[*] brown............... Brown Corpus
[*] brown_tei........... Brown Corpus (TEI XML Version)
[*] cess_cat............ CESS-CAT Treebank
[*] cess_esp............ CESS-ESP Treebank
[*] chat80.............. Chat-80 Data Files
[*] city_database....... City Database
[*] cmudict............. The Carnegie Mellon Pronouncing Dictionary (0.6)
[*] comtrans............ ComTrans Corpus Sample
[*] conll2000........... CONLL 2000 Chunking Corpus
[*] conll2002........... CONLL 2002 Named Entity Recognition Corpus
[*] conll2007........... Dependency Treebanks from CoNLL 2007 (Catalan
and Basque Subset)
[*] dependency_treebank. Dependency Parsed Treebank
[*] europarl_raw........ Sample European Parliament Proceedings Parallel
Corpus
Hit Enter to continue:
....
Downloader> d

Download which package (l=list; x=cancel)?
Identifier> all
```

If you download everything(corpora, models, grammar) NLTK needed, you can test it by running:

Downloader> u

If showing “Nothing to update”, everything is ok.

Another way to install NLTK Data is using the command, I didn’t test this way, following is from official site:

> *Python 2.5-2.7: Run the command python -m nltk.downloader all. To ensure central installation, run the command sudo python -m nltk.downloader -d /usr/share/nltk_data all.*

If you met the problem when downloading NLTK Data, such as download time out or other strange things, I suggest you download the NLTK data directly by nltk_data github page:

https://github.com/nltk/nltk_data

It said that “NLTK Data lives in the gh-pages branch of this repository.”, so you can visit the branch:

https://github.com/nltk/nltk_data/tree/master

Download the zip file and unzip it, then copy the six sub-directory in the packages into your nltk_data directory: chunkers, corpora, help, stemmers, taggers, tokenizers

Maybe this is the best unofficial way to install NLTK_Data.

Test NLTK

1) Test Brown Corpus:

```
>> from nltk.corpus import brown
>>> brown.words()[0:10]
['The', 'Fulton', 'County', 'Grand', 'Jury', 'said', 'Friday', 'an', 'investigation', 'of']
>>> brown.tagged_words()[0:10]
[('The', 'AT'), ('Fulton', 'NP-TL'), ('County', 'NN-TL'), ('Grand', 'JJ-TL'), ('Jury', 'NN-TL'),
('said', 'VBD'), ('Friday', 'NR'), ('an', 'AT'), ('investigation', 'NN'), ('of', 'IN')]
>>> len(brown.words())
```

```
1161192
>>> dir(brown)
['__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__getattribute__', '__hash__',
'__init__', '__module__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__',
'__sizeof__', '__str__', '__subclasshook__', '__weakref__', '_add', '_c2f', '_delimiter',
'_encoding', '_f2c', '_file', '_fileids', '_get_root', '_init', '_map', '_para_block_reader',
'_pattern', '_resolve', '_root', '_sent_tokenizer', '_sep', '_tag_mapping_function',
'_word_tokenizer', 'abspath', 'abspaths', 'categories', 'encoding', 'fileids', 'open', 'paras',
'raw', 'readme', 'root', 'sents', 'tagged_paras', 'tagged_sents', 'tagged_words', 'words']
```

2) Test NLTK Book Resources:

```
>>> from nltk.book import *
*** 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

>>> dir(text1)
['_CONTEXT_RE', '_COPY_TOKENS', '__class__', '__delattr__', '__dict__', '__doc__', '__format__',
'__getattribute__', '__getitem__', '__hash__', '__init__', '__len__', '__module__', '__new__',
'__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__', '__str__',
'__subclasshook__', '__weakref__', '_collocations', '_context', '_num', '_vocab', '_window_size',
'collocations', 'common_contexts', 'concordance', 'count', 'dispersion_plot', 'findall',
'generate', 'index', 'name', 'plot', 'readability', 'similar', 'tokens', 'vocab']
>>> len(text1)
260819
Sperm Whale; Moby Dick; White Whale; old man; Captain Ahab; sperm
whale; Right Whale; Captain Peleg; New Bedford; Cape Horn; cried Ahab;
```

```
years ago; lower jaw; never mind; Father Mapple; cried Stubb; chief
mate; white whale; ivory leg; one hand
```

3) Sent Tokenize(sentence boundary detection, sentence segmentation), Word Tokenize and Pos Tagging:

```
>>> from nltk import sent_tokenize, word_tokenize, pos_tag
>>> text = "Machine learning is the science of getting computers to act without being explicitly
programmed. In the past decade, machine learning has given us self-driving cars, practical speech
recognition, effective web search, and a vastly improved understanding of the human genome.
Machine learning is so pervasive today that you probably use it dozens of times a day without
knowing it. Many researchers also think it is the best way to make progress towards human-level
AI. In this class, you will learn about the most effective machine learning techniques, and gain
practice implementing them and getting them to work for yourself. More importantly, you'll learn
about not only the theoretical underpinnings of learning, but also gain the practical know-how
needed to quickly and powerfully apply these techniques to new problems. Finally, you'll learn
about some of Silicon Valley's best practices in innovation as it pertains to machine learning
and AI."
>>> sents = sent_tokenize(text)
>>> sents
['Machine learning is the science of getting computers to act without being explicitly
programmed.', 'In the past decade, machine learning has given us self-driving cars, practical
speech recognition, effective web search, and a vastly improved understanding of the human
genome.', 'Machine learning is so pervasive today that you probably use it dozens of times a day
without knowing it.', 'Many researchers also think it is the best way to make progress towards
human-level AI.', 'In this class, you will learn about the most effective machine learning
techniques, and gain practice implementing them and getting them to work for yourself.', "More
importantly, you'll learn about not only the theoretical underpinnings of learning, but also gain
the practical know-how needed to quickly and powerfully apply these techniques to new problems.",
"Finally, you'll learn about some of Silicon Valley's best practices in innovation as it pertains
to machine learning and AI."]
>>> len(sents)
7
>>> tokens = word_tokenize(text)
>>> tokens
['Machine', 'learning', 'is', 'the', 'science', 'of', 'getting', 'computers', 'to', 'act',
'without', 'being', 'explicitly', 'programmed.', 'In', 'the', 'past', 'decade', ',', 'machine',
'learning', 'has', 'given', 'us', 'self-driving', 'cars', ',', 'practical', 'speech',
```

```
'recognition', ',', 'effective', 'web', 'search', ',', 'and', 'a', 'vastly', 'improved',
'understanding', 'of', 'the', 'human', 'genome.', 'Machine', 'learning', 'is', 'so', 'pervasive',
'today', 'that', 'you', 'probably', 'use', 'it', 'dozens', 'of', 'times', 'a', 'day', 'without',
'knowing', 'it.', 'Many', 'researchers', 'also', 'think', 'it', 'is', 'the', 'best', 'way', 'to',
'make', 'progress', 'towards', 'human-level', 'AI.', 'In', 'this', 'class', ',', 'you', 'will',
'learn', 'about', 'the', 'most', 'effective', 'machine', 'learning', 'techniques', ',', 'and',
'gain', 'practice', 'implementing', 'them', 'and', 'getting', 'them', 'to', 'work', 'for',
'yourself.', 'More', 'importantly', ',', 'you', "'ll", 'learn', 'about', 'not', 'only', 'the',
'theoretical', 'underpinnings', 'of', 'learning', ',', 'but', 'also', 'gain', 'the', 'practical',
'know-how', 'needed', 'to', 'quickly', 'and', 'powerfully', 'apply', 'these', 'techniques', 'to',
'new', 'problems.', 'Finally', ',', 'you', "'ll", 'learn', 'about', 'some', 'of', 'Silicon',
'Valley', "'s", 'best', 'practices', 'in', 'innovation', 'as', 'it', 'pertains', 'to', 'machine',
'learning', 'and', 'AI', '.']
>>> len(tokens)
161
>>> tagged_tokens = pos_tag(tokens)
>>> tagged_tokens
[('Machine', 'NN'), ('learning', 'NN'), ('is', 'VBZ'), ('the', 'DT'), ('science', 'NN'), ('of',
'IN'), ('getting', 'VBG'), ('computers', 'NNS'), ('to', 'TO'), ('act', 'VB'), ('without', 'IN'),
('being', 'VBG'), ('explicitly', 'RB'), ('programmed.', 'NNP'), ('In', 'NNP'), ('the', 'DT'),
('past', 'JJ'), ('decade', 'NN'), (',', ','), ('machine', 'NN'), ('learning', 'NN'), ('has',
'VBZ'), ('given', 'VBN'), ('us', 'PRP'), ('self-driving', 'JJ'), ('cars', 'NNS'), (',', ','),
('practical', 'JJ'), ('speech', 'NN'), ('recognition', 'NN'), (',', ','), ('effective', 'JJ'),
('web', 'NN'), ('search', 'NN'), (',', ','), ('and', 'CC'), ('a', 'DT'), ('vastly', 'RB'),
('improved', 'VBN'), ('understanding', 'NN'), ('of', 'IN'), ('the', 'DT'), ('human', 'JJ'),
('genome.', 'NNP'), ('Machine', 'NNP'), ('learning', 'NN'), ('is', 'VBZ'), ('so', 'RB'),
('pervasive', 'JJ'), ('today', 'NN'), ('that', 'WDT'), ('you', 'PRP'), ('probably', 'RB'),
('use', 'VBP'), ('it', 'PRP'), ('dozens', 'VBZ'), ('of', 'IN'), ('times', 'NNS'), ('a', 'DT'),
('day', 'NN'), ('without', 'IN'), ('knowing', 'NN'), ('it.', 'NNP'), ('Many', 'NNP'),
('researchers', 'NNS'), ('also', 'RB'), ('think', 'VBP'), ('it', 'PRP'), ('is', 'VBZ'), ('the',
'DT'), ('best', 'JJS'), ('way', 'NN'), ('to', 'TO'), ('make', 'VB'), ('progress', 'NN'),
('towards', 'NNS'), ('human-level', 'JJ'), ('AI.', 'NNP'), ('In', 'NNP'), ('this', 'DT'),
('class', 'NN'), (',', ','), ('you', 'PRP'), ('will', 'MD'), ('learn', 'VB'), ('about', 'IN'),
('the', 'DT'), ('most', 'RBS'), ('effective', 'JJ'), ('machine', 'NN'), ('learning', 'NN'),
('techniques', 'NNS'), (',', ','), ('and', 'CC'), ('gain', 'NN'), ('practice', 'NN'),
('implementing', 'VBG'), ('them', 'PRP'), ('and', 'CC'), ('getting', 'VBG'), ('them', 'PRP'),
('to', 'TO'), ('work', 'VB'), ('for', 'IN'), ('yourself.', 'NNP'), ('More', 'NNP'),
('importantly', 'RB'), (',', ','), ('you', 'PRP'), ("'ll", 'MD'), ('learn', 'VB'), ('about',
```

```
'IN'), ('not', 'RB'), ('only', 'RB'), ('the', 'DT'), ('theoretical', 'JJ'), ('underpinnings',
'NNS'), ('of', 'IN'), ('learning', 'VBG'), (',', ','), ('but', 'CC'), ('also', 'RB'), ('gain',
'VBP'), ('the', 'DT'), ('practical', 'JJ'), ('know-how', 'NN'), ('needed', 'VBN'), ('to', 'TO'),
('quickly', 'RB'), ('and', 'CC'), ('powerfully', 'RB'), ('apply', 'RB'), ('these', 'DT'),
('techniques', 'NNS'), ('to', 'TO'), ('new', 'JJ'), ('problems.', 'NNP'), ('Finally', 'NNP'),
(',', ','), ('you', 'PRP'), ("'ll", 'MD'), ('learn', 'VB'), ('about', 'IN'), ('some', 'DT'),
('of', 'IN'), ('Silicon', 'NNP'), ('Valley', 'NNP'), ("'s", 'POS'), ('best', 'JJS'),
('practices', 'NNS'), ('in', 'IN'), ('innovation', 'NN'), ('as', 'IN'), ('it', 'PRP'),
('pertains', 'VBZ'), ('to', 'TO'), ('machine', 'NN'), ('learning', 'NN'), ('and', 'CC'), ('AI',
'NNP'), ('.', '.')]
```

A lot of text mining or text analysis things NLTK can do, we will introduce them in the following articles.

Posted by TextMiner

Related Posts:

1. **Dive Into NLTK, Part II: Sentence Tokenize and Word Tokenize**
2. **We have launched the Text Analysis API on Mashape**
3. **Dive Into NLTK, Part III: Part-Of-Speech Tagging and POS Tagger**
4. **Dive Into NLTK, Part VI: Add Stanford Word Segmenter Interface for Python NLTK**

This entry was posted in NLP, NLTK, Text Analysis, Text Mining and tagged Brown Corpus, Natural Language Processing, Natural Language Processing with Python, NLP, NLTK, NLTK Book, NLTK Data, NLTK Data Download, NLTK Data Install, NLTK Install, Pos Tagging, Python Natural Language Processing, Sent Tokenize, Sentence Boundary Detection, Sentence Segmentation, Text Analysis, Text Mining, TextMiner, Word Tokenize on January 17, 2014 [http://textminingonline.com/dive-into-nltk-part-i-getting-started-with-nltk] .

---

5 thoughts on “Dive Into NLTK, Part I: Getting Started with NLTK”

Pingback: Dive Into NLTK, Part II: Sentence Tokenize and Word Tokenize | Text Mining Online | Text Analysis Online

Pingback: Getting Started with Pattern | Text Mining Online | Text Analysis Online | Text Processing

Online

Pingback: Getting Started with MBSP | Text Mining Online | Text Analysis Online | Text Processing Online

Pingback: Dive Into NLTK, Part III: Part-Of-Speech Tagging and POS Tagger | Text Mining Online | Text Analysis Online | Text Processing Online

Pingback: Dive Into NLTK, Part VI: Add Stanford Word Segmenter Interface for Python NLTK | Text Mining Online | Text Analysis Online | Text Processing Online