

Information Retrieval

NLTK Tutorial

September 9, 2015

In this tutorial you will find how to install and start working with nltk to do text processing. You can find more information about NLTK in its official website www.nltk.org

1. Installing NLTK

(a) Mac/Unix

- i. Install Python: from <http://www.python.org/downloads/>
- ii. Install NLTK: run `sudo pip install -U nltk`
- iii. Install Numpy: run `sudo pip install -U numpy`

(a) Windows

- i. Install Python: from <http://www.python.org/downloads/>
- ii. Install NLTK: from <http://pypi.python.org/pypi/nltk>
- iii. Install Numpy: from <http://sourceforge.net/projects/numpy/files/NumPy/>

2. Installing NLTK Data (In python environment)

- (a) `> > > import nltk`
- (b) `> > > nltk.download()`
- (c) `> > > from nltk.corpus import brown`
- (d) `> > > brown.words()`
- (e) your output should be `['The', 'Fulton', 'County', 'Grand', 'Jury', 'said']`

3. Installing PyCharm

- (a) Download PyCharm From <https://www.jetbrains.com/pycharm/download/> and follow the instruction

4. using NLTK for text Mining

- (a) tokenizing words or sentences in a document

```
// Tokenizer.py
from nltk.tokenize import sent_tokenize, word_tokenize

sentence = " At eight o'clock on tuesday morning Mr.
            Arthur didn't feel very good. It seems that he catch a
            cold"

print(sent_tokenize(sentence))

print(word_tokenize(sentence))
```

- (b) Stemming words in a document

```
// Stemming.py

from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize

ps = PorterStemmer()

example_words = ["beautiful", "beautifuler",
                 "beautifuling", "beautifuled", "beautifully"]

for w in example_words:
    print(ps.stem(w))

sentence = "It is very important to be pythononly while you
            are pythoning with python. All pythoners have pythoned
            poorly at least once. "

words = word_tokenize(sentence)
for w in words:
```

```
print (ps.stem(w))
```

- (c) Removing stop words from a document
-

```
// StopWords.py

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

sentence = " At eight o'clock on tuesday morning Arthur
            didn't feel very good. It seems that he catch a cold"
stop_words = set(stopwords.words("english"))

# print(stop_words)

words = word_tokenize(sentence)

filtered_sentence = []

for w in words:
    if w not in stop_words:
        filtered_sentence.append(w)

print (filtered_sentence)
```

- (d) Removing punctuations from a document
-

```
// Remove-Punctuation.py
import string
sentence = " At eight o'clock on tuesday morning Arthur
            didn't feel very good. It seems that he catch a cold"

exclude = set(string.punctuation)
s = ''.join(ch for ch in sentence if ch not in exclude)
print(s)
```

- (e) using Wordnet
-

```
// wordnet.py
from nltk.corpus import wordnet
```

```
syns = wordnet.synsets("program")
print(syns)
```

(f) Tfidf

```
// In python you can use TfidfVectorizer function to find
tfidf of words in documents. If you want to use this
function you should import TfidfVectorizer from
sklearn.feature_extraction.text package
from sklearn.feature_extraction.text import TfidfVectorizer

//After that you can use TfidfVectorizer by sending your
tokenizer and stop_words

tfidf = TfidfVectorizer(tokenizer=tokenize,
                        stop_words='english')

tfs = tfidf.fit_transform(Your clean document.values())
```

(g) cosine similarity

```
// In python you can use cosine_similarity function to
find cosine similarity among different documents. If
you want to use this function you should import
cosine_similarity from sklearn.metrics.pairwise package
from sklearn.metrics.pairwise import cosine_similarity

//After that you can use cosine_similarity for your
vectorized documents
```
