NLTK 基礎功能介紹

台師大通識教育課程

文本分析與程式設計

授課：卓騰語言科技 _ PeterWolf

NLTK: Natural Language Tool Kit

- NLTK 是英文 (及其它西方語系) 自然語言處理的經典工具
- 回憶一下 NLP 進行文本前處理的幾個項目 :
  - sentence segmentation (斷句)
  - lemmatization (字型還原)
  - stopword (停用詞)
  - word segmentation (斷詞)
  - pos (詞性標記)
  - ner (命名實體辨識)
- 接著就是一些詞頻的計算，分類，模型訓練...等等的任務。

ref. http://learnenglishkids.britishcouncil.org/ja/node/14086/by_subject/2705?order=last_comment_timestamp&sort=asc&page=1%2C0%2C0%2C1

Sentence tokenize (斷句) nltk.sent_tokenize()

inputSTR01 = """I went to Japan. (NOT I went to the Japan.)
He played tennis with Ben. (NOT He played tennis with the Ben.)
They had breakfast at 9 o'clock. (NOT They had a breakfast at 9 o'clock.)
(Some words don't have an article. We don't usually use articles for countries, meals or people.)"""

inputSTR02 = """In essence, that's about as far as President Trump is likely to go. Even if he eventually acknowledges he won't be president from 20th January, he'll probably never relinquish his unsubstantiated claims that he was beaten in a fraudulent vote."""

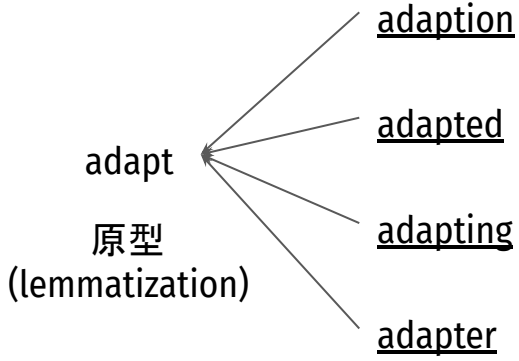
inputSTR03 = """雙 11 都過了，還在猶豫要不要入手新鍵盤嗎 ?K8 $200 折價券使用時間只到今天晚上 12 點，趕快把握時間下手最優惠！"""

```
import nltk
sentenceLIST = nltk.sent_tokenize(inputSTR01)
print(sentenceLIST)
```

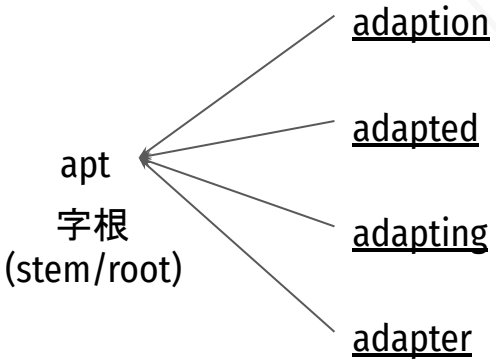
Word Lemmatization vs. Stemming

- Lemmatization => 找出「word 的原型」(沒有時態標記，沒有單複數標記)
- Stemming => 找出「word 的字根」(字根不像一個字是正常，但要挖多深？)
  - 兩種主要的 stemming 錯誤：overstemming (挖太深)、understemming (挖太淺)

理想狀況下的
lemmatization



理想狀況下的
stemming



https://ithelp.ithome.com.tw/articles/10191922

Word Lemmatization vs. Stemming

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

print(lemmatizer.lemmatize("playing", pos="v"))
```

```
LookupError:
**********************************************************************
  Resource wordnet not found.
  Please use the NLTK Downloader to obtain the resource:

  >>> import nltk
  >>> nltk.download('wordnet')

  For more information see: https://www.nltk.org/data.html

  Attempted to load corpora/wordnet

  Searched in:
    - '/Users/peter/nltk_data'
    - '/Users/peter/.pyenv/versions/3.6.9/envs/vEnv_py36/nltk_data'
    - '/Users/peter/.pyenv/versions/3.6.9/envs/vEnv_py36/share/nltk_data'
    - '/Users/peter/.pyenv/versions/3.6.9/envs/vEnv_py36/lib/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/local/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/local/lib/nltk_data'
**********************************************************************
```

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

from nltk.stem import WordNetLemmatizer
lemm = WordNetLemmatizer()

from nltk.stem import PorterStemmer
stem = PorterStemmer()

inputWord = "communication"
print(lemm.lemmatize(inputWord))
print(stem.stem(inputWord))

inputWord = "loneliness"
```

Stopwords: 停用詞 stopwords.words("english")

NLTK 的 stopwords 語料庫支援了 21 種語言，但仍以英文為主

```
from nltk.corpus import stopwords

inputSTR01 = """I went to Japan. (NOT I went to the Japan.)
He played tennis with Ben. (NOT He played tennis with the Ben.)
They had breakfast at 9 o’clock. (NOT They had a breakfast at 9 o'clock.)
(Some words don't have an article. We don't usually use articles for countries, meals or people.)"""

inputSTR01_content = ""

for i in inputSTR01:
    if i.lower() in stopwords.words("english"):
        inputSTR01_content = inputSTR01_content + "□"*len(i)
    else:
        inputSTR01_content = inputSTR01_content + i
```

```
Out[8]: "□ wen□ □□ Jopon. (N□□ □ wen□ □□ □he Jopon.)\nHe pl□□e□ □enn□□ w□□h Ben.
 (N□□ He pl□□e□ □enn□□ w□□h □he Ben.)\n□he□ h□□ breokf□□□ □□ 9 □’clock. (N□□ □he
□ h□□ □ breokf□□□ □□ 9 □'clock.)\n(□□□e w□r□□ □□n'□ h□ve □n □r□□cle. We □□n'□ u□
u□ll□ u□e □r□□cle□ for coun□r□e□, □e□l□ □r people.)"
```

Word Segmentation：斷詞 nltk.word_tokenize()

你終究還是要處理「字」的，你終究還是要「斷詞」的。

```
inputSTR01 = """I went to Japan...(下略)"""
sentenceLIST = nltk.sent_tokenize(inputSTR01)
wordLIST = []
for s in sentenceLIST:
    wordLIST.extend(nltk.word_tokenize(s))
contentLIST = []
for w in wordLIST:
    if w.lower() in stopwords.words("english"):
        contentLIST.append("□"*len(w))
    else:
        contentLIST.append(w)
print(" ".join(contentLIST))
```

```
I went to Japan. (NOT I went to the Japan.)
He played tennis with Ben. (NOT He played tennis with the Ben.)
They had breakfast at 9 o'clock. (NOT They had a breakfast at 9 o'clock.)
(Some words don't have an article. We don't usually use articles for countries,
meals or people.)
```

```
□ went □□ Japan . ( □□□ □ went □□ □□□ Japan . ) □□ played tennis □□□□ Ben . ( □□
□ □□ played tennis □□□□ □□□ Ben . ) □□□□ □□□ breakfast □□ 9 □ ' clock . ( □□□ □□
□□ □□□ □ breakfast □□ 9 o'clock . ) ( □□□□ words □□ n't □□□□ □□ article . □□ □□
n't usually use articles □□□ countries , meals □□ people . )
```

比較一下 Articut-GraphQL 的結果

重點濾鏡

周天成的逆轉秀不僅讓外媒大呼不可思議，與香港伍家朗在冠軍戰的最後一球也被BWF選為單月最佳好球。而小天連續在印尼及泰國奪冠，積分擠下中國好手石宇奇，排名升至世界男單第二，創下生涯最佳，許多外媒也看好他挾著這股氣勢，能在接下來的世錦賽中打出佳績。

Part-of-speech (POS)：詞性標記 nltk.pos_tag()

```
inputSTR01 = """I went to Japan...(下略)"""
sentenceLIST = nltk.sent_tokenize(inputSTR01)
wordLIST = []
for s in sentenceLIST:
    wordLIST.extend(nltk.word_tokenize(s))

posLIST = nltk.pos_tag(wordLIST)
print(posLIST)
```

```
In [136]: print(posLIST)
[('I', 'PRP'), ('went', 'VBD'), ('to', 'TO'), ('Japan', 'NNP'), ('.', '.'), ('('
, '('), ('NOT', 'NNP'), ('I', 'PRP'), ('went', 'VBD'), ('to', 'TO'), ('the', 'DT
'), ('Japan', 'NNP'), ('.', '.'), (')', ')'), ('He', 'PRP'), ('played', 'VBD'),
('tennis', 'NN'), ('with', 'IN'), ('Ben', 'NNP'), ('.', '.'), ('(', '('), ('NOT'
, 'NNP'), ('He', 'PRP'), ('played', 'VBD'), ('tennis', 'NN'), ('with', 'IN'), ('
the', 'DT'), ('Ben', 'NNP'), ('.', '.'), (')', ')'), ('They', 'PRP'), ('had', 'V
BD'), ('breakfast', 'NN'), ('at', 'IN'), ('9', 'CD'), ('o', 'JJ'), ("'", 'NN'),
('clock', 'NN'), ('.', '.'), ('(', '('), ('NOT', 'NNP'), ('They', 'PRP'), ('had'
, 'VBD'), ('a', 'DT'), ('breakfast', 'NN'), ('at', 'IN'), ('9', 'CD'), ("o'clock
", 'NN'), ('.', '.'), (')', ')'), ('(', '('), ('Some', 'DT'), ('words', 'NNS'),
('do', 'VBP'), ("n't", 'RB'), ('have', 'VB'), ('an', 'DT'), ('article', 'NN'), (
'.', '.'), ('We', 'PRP'), ('do', 'VBP'), ("n't", 'RB'), ('usually', 'RB'), ('use
', 'VB'), ('articles', 'NNS'), ('for', 'IN'), ('countries', 'NNS'), (',', ','),
('meals', 'NNS'), ('or', 'CC'), ('people', 'NNS'), ('.', '.'), (')', ')')]
```

https://www.ibm.com/support/knowledgecenter/zh/SS5RWK_3.5.0/com.ibm.discovery.es.ta.doc/iiysspostagset.htm

Named Entity Recognition (NER): 命名實體辨識 nltk.ne_chunk()

```
inputSTR01 = """I went to Japan...(下略)"""
sentenceLIST = nltk.sent_tokenize(inputSTR01)
wordLIST = []
for s in sentenceLIST:
    wordLIST.extend(nltk.word_tokenize(s))

posLIST = nltk.pos_tag(wordLIST)

nerLIST = nltk.ne_chunk(posLIST)
print(nerLIST)
```

```
Out[147]: Tree('S', [('I', 'PRP'), ('went', 'VBD'), ('to', 'TO'), Tree('GPE', [('
Japan', 'NNP')]), ('.', '.'), ('(', '('), ('NOT', 'NNP'), ('I', 'PRP'), ('went'
, 'VBD'), ('to', 'TO'), ('the', 'DT'), Tree('GPE', [('Japan', 'NNP')]), ('.', '.
'), (')', ')'), ('He', 'PRP'), ('played', 'VBD'), ('tennis', 'NN'), ('with', 'IN
'), Tree('PERSON', [('Ben', 'NNP')]), ('.', '.'), ('(', '('), ('NOT', 'NNP'), ('
He', 'PRP'), ('played', 'VBD'), ('tennis', 'NN'), ('with', 'IN'), ('the', 'DT'),
 Tree('ORGANIZATION', [('Ben', 'NNP')]), ('.', '.'), (')', ')'), ('They', 'PRP')
, ('had', 'VBD'), ('breakfast', 'NN'), ('at', 'IN'), ('9', 'CD'), ('o', 'JJ'), (
''', 'NN'), ('clock', 'NN'), ('.', '.'), ('(', '('), ('NOT', 'NNP'), ('They', 'P
RP'), ('had', 'VBD'), ('a', 'DT'), ('breakfast', 'NN'), ('at', 'IN'), ('9', 'CD'
), ("o'clock", 'NN'), ('.', '.'), (')', ')'), ('(', '('), ('Some', 'DT'), ('word
s', 'NNS'), ('do', 'VBP'), ("n't", 'RB'), ('have', 'VB'), ('an', 'DT'), ('articl
e', 'NN'), ('.', '.'), ('We', 'PRP'), ('do', 'VBP'), ("n't", 'RB'), ('usually',
'RB'), ('use', 'VB'), ('articles', 'NNS'), ('for', 'IN'), ('countries', 'NNS'),
(',', ','), ('meals', 'NNS'), ('or', 'CC'), ('people', 'NNS'), ('.', '.'), (')',
 ')')])
```

Quiz:

課堂中已經演示了文本處理中，利用 NLTK 進行文本分析的多種任務，包括斷句、斷詞、POS 詞性標記以及 NER 命名實體辨識和其它多種不同的 NLP 任務。試思考以下問題：

1. 如果要你設計一個斷句和斷詞的系統來處理現代英文文本，你會怎麼做？
2. 在前面幾週的課程裡，我們討論到幾種不同的斷詞系統：依規則的 Articut、依統計機率模型的 Jieba，依機器學習方法的 CKIPTagger。今天再看到預建「字典」的方法。請思考一下，NLTK 混合了哪些方法？有什麼優缺點？後續的維護怎麼處理會比較好？

Assignment: 小組作業，每組繳一份至你們的「組名目錄」即可

1. 從課程 github repo 中把課程中提供的 week10 的目錄 git pull 下來。
2. 把 week10.py 改名為 **week10_分組隊名.py**
3. 在 **week10_分組隊名.py** 中，設計你的程式，利用 NLTK 完成以下指定規格：
    a. 把 "**foxnews.json**" 中的 "content" 取出成為字串 foxnewsSTR
    b. 把 foxnewsSTR 依序斷句存成 **foxsentenceLIST**，寫回 foxnews.json 中；斷詞存成 **foxwordLIST** 寫回 foxnews.json 中；POS 處理存成 **foxPOS** 寫回 foxnews.json 中；NER 處理成 **foxNER** 寫回 foxnews.json 中。全部完成以後，請確認你的 foxnews.json 是可以被順利開啟讀取的。
    c. 英文裡的 "White House" 指的是「白宮」，但 "white house" 指的是「白色的房子」。請用 replace() 把 foxnewsSTR 中的 "White House" 改成 "white house"以後，重做步驟 b 內的操作，再觀察 NLTK 的 NER 功能用在 "White House" 和 "white house" 的辨識結果是否有所不同。