

```
!pip install nltk
import nltk
nltk.download('punkt')
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.4.2)
Requirement already satisfied: regex<=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2021.8.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.4)
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Unzipping tokenizers/punkt.zip.
True
```

```
nltk.download('gutenberg')
```

```
[nltk_data] Downloading package gutenberg to /root/nltk_data...
[nltk_data] Unzipping corpora/gutenberg.zip.
True
```

```
gb = nltk.corpus.gutenberg
print("Gutenberg files : ", gb.fileids())
```

```
Gutenberg files :  ['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt', 'bible-kjv.txt', 'blak
```

```
macbeth = nltk.corpus.gutenberg.words('shakespeare-macbeth.txt')
```

```
len(macbeth)
```

```
23140
```

```
#Nếu bạn muốn hiển thị 10 từ đầu tiên của tập tin
macbeth[:10]
```

```
['[',
 'The',
 'Tragedie',
 'of',
 'Macbeth',
 'by',
 'William',
 'Shakespeare',
 '1603',
 '']
```

```
#Hàm trên trích ra 10 từ đầu tiên là tiêu đề của tập tin, dấu ngoặc vuông được tính là 1.
#Ta muốn trích 5 câu đầu tiên của tập tin (một câu được kẹp trong cặp ngoặc vuông), ta dùng hàmsent()
macbeth_sents = nltk.corpus.gutenberg.sents('shakespeare-macbeth.txt')
macbeth_sents[:5]
```

```
[[['[',
 'The',
 'Tragedie',
 'of',
 'Macbeth',
 'by',
```

```
'William',
'Shakespeare',
'1603',
'],'],
['Actus', 'Primus', '.'],
['Scoena', 'Prima', '.'],
['Thunder', 'and', 'Lightning', '.'],
['Enter', 'three', 'Witches', '.']]
```

#2. Tìm 1 từ với NLTK:

#Tìm từ 'Stage' xuất hiện trong văn bản text

```
text = nltk.Text(macbeth)
```

```
text.concordance('Stage')
```

#Tìm từ xuất hiện trước và sau từ 'Stage'

```
text.common_contexts(['Stage'])
```

#Tìm từ tương tự từ 'Stage'

```
text.similar('Stage')
```



Displaying 3 of 3 matches:

```
nts with Dishes and Seruice ouer the Stage . Then enter Macbeth Macb . If it we
with mans Act , Threatens his bloody Stage : byth ' Clock ' tis Day , And yet d
struts and frets his houre vpon the Stage , And then is heard no more . It is
the_. bloody_: the_,
day time face warre ayre king bleeding man reuolt serieant like
knowledge broyle shew head spring heeles hare thane skie
```

#3. Phân tích tần số của các từ

```
fd = nltk.FreqDist(macbeth)
```

```
fd.most_common(10)
```



```
[(',', 1962),
('.', 1235),
(' ', 637),
('the', 531),
(':', 477),
('and', 376),
('I', 333),
('of', 315),
('to', 311),
('?', 241)]
```

#Muốn download stopwords

```
nltk.download('stopwords')
```



```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Unzipping corpora/stopwords.zip.
True
```

#Muốn xem các stopwords trong tiếng Anh, dùng lệnh

```
sw = set(nltk.corpus.stopwords.words('english'))
```

```
print(len(sw))
```

```
list(sw)[:10]
```



```
179
['mustn',
'further',
'few',
'didn',
```

```
'have',
"don't",
"isn't",
'same',
'o',
're']
```

```
#Có 179 stopword trong từ vựng tiếng Anh. Ta sẽ loại bỏ các từ stopwords trong biến macbeth
macbeth_filtered = [w for w in macbeth if w.lower() not in sw]
len(macbeth_filtered)
```

```
➞ 14946
```

```
#Kết quả như sau14946
fd = nltk.FreqDist(macbeth_filtered)
fd.most_common(10)
```

```
➞ [(',', 1962),
    ('.', 1235),
    ('"', 637),
    (':', 477),
    ('?', 241),
    ('Macb', 137),
    ('haue', 117),
    ('-', 100),
    ('Enter', 80),
    ('thou', 63)]
```

```
#
import string
punctuation = set(string.punctuation)
macbeth_filtered2 = [w.lower() for w in macbeth if w.lower() not in sw and \
                    w.lower() not in punctuation]
fd = nltk.FreqDist(macbeth_filtered2)
fd.most_common(10)
```

```
➞ [('macb', 137),
    ('haue', 122),
    ('thou', 90),
    ('enter', 81),
    ('shall', 68),
    ('macbeth', 62),
    ('vpon', 62),
    ('thee', 61),
    ('macd', 58),
    ('vs', 57)]
```

```
#4. Lựa chọn các từ trong văn bản
long_words = [w for w in macbeth if len(w)> 12]
sorted(long_words)
```

```
➞ ['Assassination',
    'Chamberlaines',
    'Distinguishes',
    'Gallowgrosses',
    'Metaphysicall',
    'Northumberland',
```

```
'Voluptuousnesse',
'commendations',
'multitudinous',
'supernaturall',
'vnaccompanied']
```

```
##Rút trích các từ có chứa chuỗi 'ious'
ious_words = [w for w in macbeth if 'ious' in w]
ious_words = set(ious_words)
sorted(ious_words)
```

```
➡ ['Auaricious',
'Gracious',
'Industrious',
'Iudicious',
'Luxurious',
'Malicious',
'Obliuious',
'Pious',
'Rebellious',
'compunctious',
'furious',
'gracious',
'pernicious',
'pernitious',
'pious',
'precious',
'rebellious',
'sacrilegious',
'serious',
'spacious',
'tedious']
```

#5. Bigrams và collocations

#Lọc các bigram sau khi đã loại các stopwords và các dấu câu, dùng lệnh sau:

```
bgrms = nltk.FreqDist(nltk.bigrams(macbeth_filtered2))
bgrms.most_common(15)
```

```
➡ [ (('enter', 'macbeth'), 16),
 (('exeunt', 'scena'), 15),
 (('thane', 'cawdor'), 13),
 (('knock', 'knock'), 10),
 (('st', 'thou'), 9),
 (('thou', 'art'), 9),
 (('lord', 'macb'), 9),
 (('haue', 'done'), 8),
 (('macb', 'haue'), 8),
 (('good', 'lord'), 8),
 (('let', 'vs'), 7),
 (('enter', 'lady'), 7),
 (('wee', 'l'), 7),
 (('would', 'st'), 6),
 (('macbeth', 'macb'), 6)]
```

#Ngoài bigram ra, còn có trigram, sự kết hợp của 3 từ, ta dùng lệnh trigrams()

```
tgrms = nltk.FreqDist(nltk.trigrams (macbeth_filtered2))
tgrms.most_common(10)
```

```
➡ [ (('knock', 'knock', 'knock'), 6),
 (('enter', 'macbeth', 'macb'), 5),
 (('enter', 'three', 'witches'), 4),
 (('exeunt', 'scena', 'secunda'), 4),
 (('good', 'lord', 'macb'), 4),
```

```
((('three', 'witches', '1'), 3),
 (('exeunt', 'scena', 'tertia'), 3),
 (('thunder', 'enter', 'three'), 3),
 (('exeunt', 'scena', 'quarta'), 3),
 (('scena', 'prima', 'enter'), 3])
```

#6. Sử dụng văn bản trên mạng

#Import thư viện và mở url để đọc file

```
from urllib import request
url = "http://www.gutenberg.org/files/2554/2554-0.txt"
response = request.urlopen(url)
raw = response.read().decode('utf8')
raw[:75]
```

```
↩ '\uffffThe Project Gutenberg eBook of Crime and Punishment, by Fyodor Dostoevsky\r'
```

#Thay bằng các lệnh sau:

```
from urllib import request
url = "http://www.gutenberg.org/files/2554/2554-0.txt"
response = request.urlopen(url)
# Try using 'utf-8' encoding instead of 'utf8-sig'
raw = response.read().decode('utf-8')
raw[:75]
```

```
↩ '\uffffThe Project Gutenberg eBook of Crime and Punishment, by Fyodor Dostoevsky\r'
```

#Thực hiện các lệnh sau:

```
tokens = nltk.word_tokenize (raw)
webtext = nltk.Text (tokens)
webtext[:12]
```

```
↩ ['\uffffThe',
 'Project',
 'Gutenberg',
 'eBook',
 'of',
 'Crime',
 'and',
 'Punishment',
 ',',
 'by',
 'Fyodor',
 'Dostoevsky']
```

#7. Rút trích văn bản từ trang html

```
url = "http://news.bbc.co.uk/2/hi/health/2284783.stm"
html = request.urlopen(url).read().decode('utf8')
html[:120]
```

```
from bs4 import BeautifulSoup
raw = BeautifulSoup(html, "lxml").get_text()
tokens = nltk.word_tokenize(raw)
text = nltk.Text(tokens)
```

#8. Phân tích cảm xúc người dùng

#Trước tiên, download các movie review dùng lệnh sau:


```
nltk.download('movie_reviews')
```

```

import random
reviews = nltk.corpus.movie_reviews
documents = [(list(reviews.words(fileid)), category)
for category in reviews.categories()
for fileid in reviews.fileids(category)]
random.shuffle(documents)

#Xem nội dung review đầu tiên (dòng 0, cột 0)
first_review = ' '.join(documents[0][0])
print(first_review)

```

 [nltk_data] Downloading package movie_reviews to /root/nltk_data...
 [nltk_data] Unzipping corpora/movie_reviews.zip.
 just look back two years ago at the coen brothers ' comedic gem the big lebowski , change the actors , tak

```

#Xem kết quả review đầu tiên (dòng 0, cột 1)
documents[0][1]

```

 'pos'

```

#Ta cần tạo bảng phân phối tần số các từ trong corpus, bảng này cần chuyển sang dạng list, ta dùng
#hàm list()
all_words = nltk.FreqDist(w.lower() for w in reviews.words())
word_features = list(all_words)

```

```

#Sau đó, bước tiếp theo là xác định một hàm để tính toán các đặc trưng, tức là những từ đủ quan
#trọng để thiết lập ý kiến của một review.
def document_features(document, word_features):
    document_words = set(document)
    features = {}
    for word in word_features:
        features['{}'.format(word)] = (word in document_words)
    return features

```

```

#Khi bạn định nghĩa hàm document_features(), bạn tạo 1 tập các documents
featuresets = [(document_features(d,word_features), c) for (d,c) in documents]
len(featuresets)

```

 2000

```


#Tạo tập train và tập test: 1500 dòng đầu dùng cho tập train và 500 dòng còn lại dùng cho tập test
#để đánh giá độ chính xác của mô hình.
train_set, test_set = featuresets[1500:], featuresets[:500]
classifier = nltk.NaiveBayesClassifier.train(train_set)

```

```

#Dùng thuật toán Naïve Bayes để phân loại, dùng thư viện NLTK. Sau đó tính toán độ chính xác
#của thuật toán
train_set, est_set = featuresets[1500:], featuresets[:500]
classifier = nltk.NaiveBayesClassifier.train(train_set)
print(nltk.classify.accuracy(classifier, test_set))

```

 0.742

```
classifier.show_most_informative_features(10)
```



Most Informative Features

festival = True	pos : neg = 10.0 : 1.0
multi = True	pos : neg = 10.0 : 1.0
chilling = True	pos : neg = 9.2 : 1.0
outstanding = True	pos : neg = 8.7 : 1.0
stunning = True	pos : neg = 8.7 : 1.0
terrific = True	pos : neg = 8.7 : 1.0
brilliant = True	pos : neg = 8.5 : 1.0
belief = True	pos : neg = 8.5 : 1.0
contemporary = True	pos : neg = 8.5 : 1.0
bore = True	neg : pos = 8.1 : 1.0