LAB 7: PHÂN TÍCH DỮ LIỆU DẠNG VĂN BẢN VỚI NLTK

1. Giới thiệu về thư viện NLTK:

```
import nltk
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('maxent_ne_chunker')
nltk.download('words')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping chunkers/maxent_ne_chunker.zip.
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Unzipping corpora/words.zip.
True
```

```
[2] 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', 'blake-poems.txt', 'bryant-stories.txt', 'burgess-busterbrown.txt', 'carroll-alice.
```

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

```
[5] len(macbeth)
```

```
23140
```

```
macbeth [:10]
```

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

```
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:

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

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

```
the_. bloody_: the_,
```

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

```
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)]
```

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

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

```
sw = set(nltk.corpus.stopwords.words('english'))
print(len(sw))
list(sw)[:10]
```

```
198
['ll',
 'are',
 "didn't",
 "you'd",
 'yourself',
 'own',
 'm',
 'have',
 'didn',
 "hasn't"]
```

```
[16] macbeth_filtered = [w for w in macbeth if w.lower() not in sw]
     len(macbeth_filtered)
```

```
14946
```

```
fd = nltk.FreqDist(macbeth_filtered)
fd.most_common(10)
```

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

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

     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

```
[20] 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']
```

```
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

```
[22] 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)]
```

```
[23] 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

```
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]
```

```
'*** START OF THE PROJECT GUTENBERG EBOOK 2554 ***\n\n\n\n\nCRIME AND PUNISHMENT
\n'
```

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

```
'*** START OF THE PROJECT GUTENBERG EBOOK 2554 ***\n\n\n\n\nCRIME AND PUNISHMENT
\n'
```

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

```
['*',
 '*',
 '*',
 'START',
 'OF',
 'THE',
 'PROJECT',
 'GUTENBERG',
 'EBOOK',
 '2554',
 '*',
 '*']
```

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]
```

```
'<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN" "http://www.w3.org/T
R/REC-html40/loose.dtd">\r\n<html>\r\n<hea'
```

```
[30] 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

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

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

```
[34] 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)
```

```
first_review = ' '.join(documents[0][0])
print(first_review)
```

```
surrounded by hype , high hopes , and the promise of an over - the - top performance
```

```
documents[0][1]
```

```
'neg'
```

```
[37] all_words = nltk.FreqDist(w.lower() for w in reviews.words())
     word_features = list(all_words)
```

```
[38] 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
```

```
[41] featuresets = [(document_features(d,word_features), c) for (d,c) in documents]
     len(featuresets)
```

```
2000
```

```
[42] train_set, test_set = featuresets[1500:], featuresets[:500]
     classifier = nltk.NaiveBayesClassifier.train(train_set)
```

```
train_set, est_set = featuresets[1500:], featuresets[:500]
classifier = nltk.NaiveBayesClassifier.train(train_set)
print(nltk.classify.accuracy(classifier, test_set))
```

```
0.788
```

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

```
Most Informative Features
                portrayal = True              pos : neg    =     18.2 : 1.0
             uninteresting = True              neg : pos    =     10.0 : 1.0
                      zero = True              neg : pos    =      9.4 : 1.0
                      lame = True              neg : pos    =      8.7 : 1.0
             controversial = True              pos : neg    =      8.6 : 1.0
                      bore = True              neg : pos    =      8.1 : 1.0
                      anna = True              pos : neg    =      7.9 : 1.0
                   freedom = True              pos : neg    =      7.9 : 1.0
                remembered = True              pos : neg    =      7.9 : 1.0
                    bother = True              neg : pos    =      7.4 : 1.0
```

9. Bài tập áp dụng

- Cài các thư viện cần thiết

```python
import nltk
from nltk.corpus import stopwords, wordnet, names
from nltk.tokenize import word_tokenize
from nltk.corpus import wordnet as wn
from nltk.tag import pos_tag
import random
nltk.download('all')
```

```
[nltk_data] Downloading collection 'all'
[nltk_data]    |
[nltk_data]    | Downloading package abc to /root/nltk_data...
[nltk_data]    |   Package abc is already up-to-date!
[nltk_data]    | Downloading package alpino to /root/nltk_data...
[nltk_data]    |   Package alpino is already up-to-date!
[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger is already up-
[nltk_data]    |       to-date!
[nltk_data]    | Downloading package averaged_perceptron_tagger_eng to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger_eng is already
[nltk_data]    |       up-to-date!
[nltk_data]    | Downloading package averaged_perceptron_tagger_ru to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger_ru is already
[nltk_data]    |       up-to-date!
[nltk_data]    | Downloading package averaged_perceptron_tagger_rus to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger_rus is already
[nltk_data]    |       up-to-date!
[nltk_data]    | Downloading package basque_grammars to
```

1. Viết chương trình Python với thư viện NLTK để liệt kê các tên của copus.

```python
import os
from nltk.data import find
corpus_path = find("corpora")
print(os.listdir(corpus_path))
```

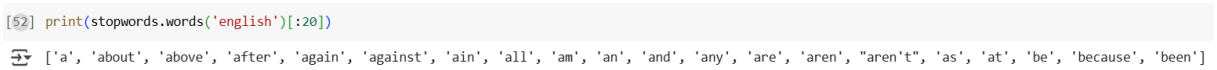
```
['toolbox', 'dependency_treebank', 'abc.zip', 'udhr.zip', 'paradigms', 'kimmo.zip', 'floresta', 'english_wordnet', 'indian', 'framenet_v15.zip', 'crubadan.zip', 'wordnet_ic', 'shakesp
```

2. Viết chương trình Python với thư viện NLTK để liệt kê danh sách các stopword bằng các ngôn ngữ khác nhau.

```python
[51] nltk.download('stopwords')
     print(stopwords.fileids())
```

```
['albanian', 'arabic', 'azerbaijani', 'basque', 'belarusian', 'bengali', 'catalan', 'chinese', 'danish', 'dutch', 'english', 'finnish', 'french', 'german', 'greek', 'hebrew', 'hinglish
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

3. Viết chương trình Python với thư viện NLTK để kiểm tra danh sách các stopword bằng các ngôn ngữ khác nhau.



```python
[52] print(stopwords.words('english')[:20])
```

```
['a', 'about', 'above', 'after', 'again', 'against', 'ain', 'all', 'am', 'an', 'and', 'any', 'are', 'aren', "aren't", 'as', 'at', 'be', 'because', 'been']
```

4. Viết chương trình Python với thư viện NLTK để loại bỏ các stopword từ một văn bản đã cho.

```
nltk.download('punkt')
text = "This is a sample sentence, showing off the stop words filtration."
words = word_tokenize(text)
filtered = [w for w in words if w.lower() not in stopwords.words('english')]
print(filtered)
```

```
['sample', 'sentence', ',', 'showing', 'stop', 'words', 'filtration', '.']
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

5. Viết chương trình Python với thư viện NLTK bỏ qua các stopword từ danh sách các stopword.

```
[57] om_list = ['i', 'am', 'learning', 'nltk', 'and',' it', 'is','awesome']
     ered_custom = [w for w in custom_list if w not in stopwords.words('english')]
     t(filtered_custom)
```

```
['learning', 'nltk', ' it', 'awesome']
```

6. Viết một chương trình Python với thư viện NLTK để tìm định nghĩa và ví dụ của một từ đã cho bằng WordNet từ Wikipedia,

```
[60] nltk.download('wordnet')
     word = 'computer'
     syns = wn.synsets(word)
     for s in syns[:1]:
       print("Định nghĩa: ", s.definition())
       print("Ví dụ: ", s.examples())
```

```
Định nghĩa:  a machine for performing calculations automatically
Ví dụ:  []
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

7. Viết chương trình Python với thư viện NLTK để tìm tập hợp các từ đồng nghĩa và trái nghĩa của một từ nào đó.

```
synonyms = set()
antonyms = set()
for syn in wordnet.synsets("good"):
    for l in syn.lemmas():
        synonyms.add(l.name())
        if l.antonyms():
            antonyms.add(l.antonyms()[0].name())
print("Đồng nghĩa của 'good': ", list(synonyms)[:10])
print("Trái nghĩa của 'good': ", list(antonyms))
```

```
Đồng nghĩa của 'good':  ['serious', 'unspoilt', 'in_effect', 'thoroughly', 'secure', 'expert', 'well', 'unspoiled', 'good', 'trade_good']
Trái nghĩa của 'good':  ['badness', 'evil', 'bad', 'evilness', 'ill']
```

8. Viết chương trình Python với thư viện NLTK để có cái nhìn tổng quan về bộ tag, chi tiết của một tag cụ thể trong bộ tag và chi tiết về một số bộ tag liên quan, sử dụng biểu thức chính quy.

```
[62] nltk.download('brown')
     nltk.download('tagsets')
     nltk.help.upenn_tagset('NN')
```

```
NN: noun, common, singular or mass
    common-carrier cabbage knuckle-duster Casino afghan shed thermostat
    investment slide humour falloff slick wind hyena override subhumanity
    machinist ...
[nltk_data] Downloading package brown to /root/nltk_data...
[nltk_data]   Package brown is already up-to-date!
[nltk_data] Downloading package tagsets to /root/nltk_data...
[nltk_data]   Package tagsets is already up-to-date!
```

9. Viết chương trình Python với thư viện NLTK để so sánh sự giống nhau của hai danh từ đã cho.

```
[63] w1 = wordnet.synset('car.n.01')
     w2 = wordnet.synset('bus.n.01')
     print("Mức độ tương đồng giữa car và bus: ", w1.wup_similarity(w2))
```

```
Mức độ tương đồng giữa car và bus:  0.6666666666666666
```

10. Viết chương trình Python với thư viện NLTK để so sánh sự giống nhau của hai động từ đã cho.

```
[64] v1 = wordnet.synset('run.v.01')
     v2 = wordnet.synset('walk.v.01')
     print("Mức độ tương đồng giữa run và walk: ", v1.wup_similarity(v2))
```

```
Mức độ tương đồng giữa run và walk:  0.2857142857142857
```

11. Viết chương trình Python với thư viện NLTK để tìm số lượng tên nam và nữ trong các tên kho ngữ liệu. In tên 10 nam và nữ đầu tiên. Lưu ý: Kho văn bản tên chứa tổng cộng khoảng 2943 nam (male.txt) và 5001 nữ (Female.txt) tên. Kho được biên soạn bởi Kantrowitz, Ross.

```
[65] nltk.download('names')
     male_names = names.words('male.txt')
     female_names = names.words('female.txt')
     print("Số lượng tên nam: ", len(male_names))
     print("Số lượng tên nữ: ", len(female_names))
     print("10 tên nam đầu tiên: ", male_names[:10])
     print("10 tên nữ đầu tiên: ", female_names[:10])
```

```
Số lượng tên nam:  2943
Số lượng tên nữ:  5001
10 tên nam đầu tiên:  ['Aamir', 'Aaron', 'Abbey', 'Abbie', 'Abbot', 'Abbott', 'Abby', 'Abdel', 'Abdul', 'Abdulkarim']
10 tên nữ đầu tiên:  ['Abagael', 'Abagail', 'Abbe', 'Abbey', 'Abbi', 'Abbie', 'Abby', 'Abigael', 'Abigail', 'Abigale']
[nltk_data] Downloading package names to /root/nltk_data...
[nltk_data]   Package names is already up-to-date!
```

12. Viết chương trình Python với thư viện NLTK để in 15 kết hợp ngẫu nhiên đầu tiên được gắn nhãn nam và được gắn nhãn tên nữ từ kho tên.

```
[67] labled_names = ([(name, 'male') for name in male_names] + [(name, 'female') for name in female_names])
     random.shuffle(labled_names)
     print("15 tên ngẫu nhiên (nam/nữ)")
     print(labled_names[:15])
```

```
15 tên ngẫu nhiên (nam/nữ)
[('Wake', 'male'), ('Pavel', 'male'), ('Pavla', 'female'), ('Cordelia', 'female'), ('Vale', 'male'), ('Arly', 'female'), ('Garcia', 'male'), ('Franni', 'female'), ('Hanni', 'female'),
```

13. Viết chương trình Python với thư viện NLTK để trích xuất ký tự cuối cùng của tất cả các tên được gắn nhãn và tạo mảng mới với chữ cái cuối cùng của mỗi tên và nhãn được liên kết

```
[68] features = [(name[-1], gender) for (name, gender) in labled_names]
     print("Mảng (ký tự cuối, nhãn)")
     print(features[:15])
```

```
Mảng (ký tự cuối, nhãn)
[('e', 'male'), ('l', 'male'), ('a', 'female'), ('a', 'female'), ('e', 'male'), ('y', 'female'), ('a', 'male'), ('i', 'female'), ('i', 'female'), ('a', 'female'), ('e', 'female'), ('e'
```