```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os


import nltk


from nltk.corpus import product_reviews_1


nltk.download('product_reviews_1')
```

```
[nltk_data] Downloading package product_reviews_1 to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping corpora/product_reviews_1.zip.
True
```

```
product_reviews_1.fileids()
```

```
['Apex_AD2600_Progressive_scan_DVD player.txt',
 'Canon_G3.txt',
 'Creative_Labs_Nomad_Jukebox_Zen_Xtra_40GB.txt',
 'Nikon_coolpix_4300.txt',
 'Nokia_6610.txt',
 'README.txt']
```

```
product_review_raw = product_reviews_1.raw('Apex_AD2600_Progressive_scan_DVD player
product_review_raw[:750]
```

```
'**********************************************************************
*\n* Annotated by: Minqing Hu and Bing Liu, 2004.\n*\t\tDepartment of Compute
r Sicence\n*              University of Illinois at Chicago             \n*
\n* Product name: Apex AD2600 Progressive-scan DVD player\n* Review Source: a
mazon.com\n*\n* See Readme.txt to find the meaning of each symbol. \n********
**********************************************************************\n\n[t]
troubleshooting ad-2500 and ad-2600 no picture scrolling b/w . \n##repost fro
m january 13 , 2004 with a better fit title . \n##does your apex dvd player o
```

```
product_review_sents = product_reviews_1.sents('Apex_AD2600_Progressive_scan_DVD pl
product_review_sents
```

```
[['repost', 'from', 'january', '13', ',', '2004', 'with', 'a', 'better', 'fit'
```

```
product_review_words = product_reviews_1.words('Apex_AD2600_Progressive_scan_DVD pl
product_review_words
```

```
['repost', 'from', 'january', '13', ',', '2004', ...]
```

```
product_review_words = product_reviews_1.words('Apex_AD2600_Progressive_scan_DVD pl
product_review_words
```

```
['repost', 'from', 'january', '13', ',', '2004', ...]
```

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

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

```
from nltk.corpus import stopwords
stoplist = stopwords.words('english')
print(stoplist)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're",
```

```
print(f'word length with stopwords {len(product_review_words)}')
product_review_wo_stopwords = [word for word in product_review_words if not word in
print(f'word length without stopwords {len(product_review_wo_stopwords)}')
```

```
word length with stopwords 12593
word length without stopwords 7190
```

```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
from nltk.tokenize import sent_tokenize, word_tokenize
```

```
print(f'Word Tokens - \n{sent_tokenize(product_review_raw[750:1250])}\n\n\n')
print(f'Sentence Tokens - \n{word_tokenize(product_review_raw[750:1250])}')
```

```
Word Tokens -
['te hours calling apex tech support , or run the player over with your car ,
```

```
Sentence Tokens -
['te', 'hours', 'calling', 'apex', 'tech', 'support', ',', 'or', 'run', 'the',
```

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

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

```
from nltk.stem import WordNetLemmatizer
from nltk.stem.porter import PorterStemmer
sample_sentence = 'A middle-aged woman entered the room, her hands full of hamburge
porter_stemmer = PorterStemmer()
word_lemmatizer = WordNetLemmatizer()
```

```
for w in word_tokenize(sample_sentence):
```

```
print(f'Actual Word - {w}')
print(f'Stem - {porter_stemmer.stem(w)}')
print(f'Lemma - {word_lemmatizer.lemmatize(w)}\n')
```

```
Actual Word - A
Stem - A
Lemma - A

Actual Word - middle-aged
Stem - middle-ag
Lemma - middle-aged

Actual Word - woman
Stem - woman
Lemma - woman

Actual Word - entered
Stem - enter
Lemma - entered

Actual Word - the
Stem - the
Lemma - the

Actual Word - room
Stem - room
Lemma - room

Actual Word - ,
Stem - ,
Lemma - ,

Actual Word - her
Stem - her
Lemma - her

Actual Word - hands
Stem - hand
Lemma - hand

Actual Word - full
Stem - full
Lemma - full

Actual Word - of
Stem - of
Lemma - of

Actual Word - hamburger
Stem - hamburg
Lemma - hamburger

Actual Word - meat
Stem - meat
Lemma - meat

Actual Word - as
Stem - as
Lemma - a

Actual Word - she
```

```
Stem - she
Lemma - she
```

```
from sklearn.feature_extraction.text import CountVectorizer
# initialize sample document
sample_documents = ['Must have a subject and a verb','Must express a complete thoug
# instantiate
vectorizer = CountVectorizer()
vectorizer.fit(sample_documents)
# summarize
print(f':: vector vocabulary - {vectorizer.vocabulary_}\n')
# encode document
vector = vectorizer.transform(sample_documents)
# summarize encoded vector
print(f':: vector shape - {vector.shape}\n')
print(f':: vector list - {vector.toarray()}')
```

```
:: vector vocabulary - {'must': 5, 'have': 4, 'subject': 8, 'and': 0, 'verb':

:: vector shape - (3, 11)

:: vector list - [[1 0 0 0 1 1 0 0 1 0 1]
 [0 0 1 1 0 1 0 0 0 1 0]
 [0 1 0 0 1 1 1 1 0 0 0]]
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
# initialize sample document
sample_documents = ['Must have a subject and a verb','Must express a complete thoug
# instantiate
vectorizer = TfidfVectorizer()
vectorizer.fit(sample_documents)
# summarize
print(f':: vector vocabulary - {vectorizer.vocabulary_}\n')
# encode document
vector = vectorizer.transform(sample_documents)
# summarize encoded vector
print(f':: vector shape - {vector.shape}\n')
print(f':: vector list - {vector.toarray()}')
```

```
:: vector vocabulary - {'must': 5, 'have': 4, 'subject': 8, 'and': 0, 'verb':

:: vector shape - (3, 11)

:: vector list - [[0.50461134 0.         0.         0.         0.38376993 0.29
  0.         0.         0.50461134 0.         0.50461134]
 [0.         0.         0.54645401 0.54645401 0.         0.32274454
  0.         0.         0.         0.54645401 0.        ]
 [0.         0.50461134 0.         0.         0.38376993 0.29803159
  0.50461134 0.50461134 0.         0.         0.        ]]
```