

```
In [1]: import nltk
```

```
In [2]: nltk.download()
```

showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
1 (https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml)

```
Out[2]: True
```

```
In [3]: paragraph = "He is a good boy. She is a good girl. boy & girl are good."
```

Tokenization

Sentence tokenization

```
In [4]: sentences = nltk.sent_tokenize(paragraph)
sentences
```

```
Out[4]: ['He is a good boy.', 'She is a good girl.', 'boy & girl are good.']
```

word tokenization

```
In [5]: words = nltk.word_tokenize(paragraph)
words
```

```
Out[5]: ['He',
'is',
'a',
'good',
'boy',
'.',
'She',
'is',
'a',
'good',
'girl',
'.',
'boy',
'&',
'girl',
'are',
'good',
'.']
```

Text Cleaning

remove punctuation

In [6]: `import re`

```
corpus=[]
for i in range(len(sentences)):
    rp = re.sub('[^a-zA-Z]', " ", sentences[i])
    corpus.append(rp)

print(corpus)
```

`['He is a good boy ', 'She is a good girl ', 'boy girl are good ']`

List of stopwords in English

In [7]: `from nltk.corpus import stopwords`
`stopwords.words('english')`

...

Remove stopwords

In [8]: `corpus=[]`
`for i in range(len(sentences)):`
 `rp = re.sub('[^a-zA-Z]', " ", sentences[i])`
 `rp = rp.lower()`
 `rp = rp.split()`
 `rp = [word for word in rp if not word in set(stopwords.words('english'))]`
 `rp = " ".join(rp)`
 `corpus.append(rp)`

`print(corpus)`

`['good boy', 'good girl', 'boy girl good']`

Stemming

In [9]: `from nltk.stem import PorterStemmer`
`ps = PorterStemmer()`
`ps.stem("history")`

Out[9]: `'histori'`

Lemmatization

In [10]: `from nltk.stem import WordNetLemmatizer`
`wnl = WordNetLemmatizer()`
`wnl.lemmatize("history")`

Out[10]: `'history'`

Text Cleaning (Remove punctuation + Remove Stopwords + Stemming/ Lemmatization)

```
In [11]: corpus=[]
for i in range(len(sentences)):
    rp = re.sub('[^a-zA-Z]', " ", sentences[i])
    rp = rp.lower()
    rp = rp.split()
    rp = [ps.stem(word) for word in rp if not word in set(stopwords.words('english'))]
    rp = " ".join(rp)
    corpus.append(rp)
```

Vectorization

Count Vectorizer (Bag of Words)

```
In [12]: from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
bow = cv.fit_transform(corpus).toarray()
bow
```

```
Out[12]: array([[1, 0, 1],
                [0, 1, 1],
                [1, 1, 1]], dtype=int64)
```

TF-IDF Vectorizer

```
In [13]: from sklearn.feature_extraction.text import TfidfVectorizer
tf = TfidfVectorizer()
tfidf = tf.fit_transform(corpus).toarray()
tfidf
```

```
Out[13]: array([[0.78980693, 0.61335554],
                [0.78980693, 0.61335554],
                [0.61980538, 0.61980538, 0.48133417]])
```

Part of Speech Tagging (POS Tagging)

```
In [14]: text = "I love natural Language Processing"

words = nltk.word_tokenize(text)

nltk.pos_tag(words)
```

```
Out[14]: [('I', 'PRP'),
          ('love', 'VBP'),
          ('natural', 'JJ'),
          ('Language', 'NNP'),
          ('Processing', 'NN')]
```

```
In [15]: nltk.help.upenn_tagset("PRP")
```

PRP: pronoun, personal

hers herself him himself hisself it itself me myself one oneself ours
ourselves ownself self she thee theirs them themselves they thou thy us

"Data Science & AI"
by
Siva Rama Krishna