NLTK (Natural Language Toolkit) is a set of Python modules to carry out many common Natural Language Processing (NLP) tasks.

NLTK provides:

- Basic classes for representing data relevent to NLP.
- Standard interfaces for performing tasks, such as tokenization, stemming, lemmetization.
- Standard implementation of each task, which can be combined to solve complex problems.

Installing NLTK

Prerequisite: must have Python installed.

To install NLTK library, open the command terminal and type:

```
pip install nltk
```

```
!pip install nltk
```

```
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) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.4)
```

```
import nltk
nltk.download('punkt')      # download resources for tokenization & puntuations
nltk.download('stopwords')  # download resources for stopwords
nltk.download('wordnet')    # download resources for lemmetization

# nltk.download('all')      # download all resourses
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
True
```

Tokenization

The process of breaking down text paragraphs into smaller chunks such as words or sentences is called **Tokenization**.

- Sentence Tokenization: breaks text paragraph into sentences.
- Word Tokenization: breaks text paragraphs into words.

```
# file = open('data.txt', 'r')
# text = file.read()

text = "Mr. Smith is feeling Relaxed today, as The weather in USA is awesome. Did something troubled Him in U.S.A.? The birds are flying
```

```
# Sentence Tokenization
from nltk.tokenize import sent_tokenize

tokenized_sent = sent_tokenize(text)
print(tokenized_sent)
```

```
['Mr. Smith is feeling Relaxed today, as The weather in USA is awesome.', 'Did something troubled Him in U.S.A.?', 'The birds are fl
```

```
# Word Tokenization
from nltk.tokenize import word_tokenize

tokenized_word = word_tokenize(text)
print(tokenized_word)
```

```
['Mr.', 'Smith', 'is', 'feeling', 'Relaxed', 'today', ',', 'as', 'The', 'weather', 'in', 'USA', 'is', 'awesome', '.', 'Did', 'someth
```

Frequency Distribution

```
from nltk.probability import FreqDist

fdist = FreqDist(tokenized_word)
print(fdist.most_common(2))
```

```
[('is', 2), ('The', 2)]
```

Case Folding

- convert all to Lower-case
- use True-casing

```
# lowercasing
lower_token = []
for token in tokenized_word:
    lower_token.append(token.lower())

print(lower_token)
```

```
['mr.', 'smith', 'is', 'feeling', 'relaxed', 'today', ',', 'as', 'the', 'weather', 'in', 'usa', 'is', 'awesome', '.', 'did', 'someth
```

```
# installing truecase
!pip install truecase
from truecase import get_true_case
```

```
Requirement already satisfied: truecase in /usr/local/lib/python3.10/dist-packages (0.0.14)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (from truecase) (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk->truecase) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk->truecase) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk->truecase) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk->truecase) (4.66.4)
```

```
# truecase sentences
true_text = []
for text in tokenized_sent:
    true_text.append(get_true_case(text))
print(true_text)
```

```
['Mr. Smith is feeling relaxed today, as the weather in USA is awesome.', 'Did something troubled him in U.S.A.?', 'The birds are fl
```

```
# tokenize truecase words
true_token = []
for token in true_text:
    true_token.extend(word_tokenize(token))
print(true_token)
```

```
['Mr.', 'Smith', 'is', 'feeling', 'relaxed', 'today', ',', 'as', 'the', 'weather', 'in', 'USA', 'is', 'awesome', '.', 'Did', 'someth
```

```
import pandas as pd
d = {'Original': tokenized_word, 'Lowercase': lower_token, 'Truecase': true_token}
df = pd.DataFrame(d)
# df['Lowercase'] = lower_token
# df['Truecase'] = true_token

df
```

	Original	Lowercase	Truecase
0	Mr.	mr.	Mr.
1	Smith	smith	Smith
2	is	is	is
3	feeling	feeling	feeling
4	Relaxed	relaxed	relaxed
5	today	today	today
6	,	,	,
7	as	as	as
8	The	the	the
9	weather	weather	weather
10	in	in	in
11	USA	usa	USA
12	is	is	is
13	awesome	awesome	awesome
14	.	.	.
15	Did	did	Did
16	something	something	something
17	troubled	troubled	troubled
18	Him	him	him
19	in	in	in
20	U.S.A.	u.s.a.	U.S.A.
21	?	?	?
22	The	the	The
23	birds	birds	birds
24	are	are	are
25	flying	flying	flying
26	.	.	.

Next steps: Generate code with df | View recommended plots

Removing Punctuations

```
import string

# punctuations
punctuations = list(string.punctuation)

tokens = []

for i in lower_token:
    if i not in punctuations:
        tokens.append(i)

print(tokens)
```

```
['mr.', 'smith', 'is', 'feeling', 'relaxed', 'today', 'as', 'the', 'weather', 'in', 'usa', 'is', 'awesome', 'did', 'something', 'tro
```

Stopwords

Stopwords are considered noise in text. Text may contain stop words such as is, am, are, this, a, an, the, etc.

```
from nltk.corpus import stopwords

stopwords_english = stopwords.words("english")
print(stopwords_english)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourse
```

In NLTK for removing stopwords, you need to create a list of stopwords and filter out your list of tokens from these words.

```
# Removing Stopwords

filtered_tokens=[]
for w in tokens:
    if w not in stopwords_english:
        filtered_tokens.append(w)
print(filtered_tokens)
```

```
['mr.', 'smith', 'feeling', 'relaxed', 'today', 'weather', 'usa', 'awesome', 'something', 'troubled', 'u.s.a.', 'birds', 'flying']
```

Stemming and Lemmetization

Stemming is a process of linguistic normalization, which reduces words to their word root or chops off the derivational affixes.

```
# Stemming
from nltk.stem import PorterStemmer
ps = PorterStemmer()

stem_words=[]

for w in filtered_tokens:
     stem_words.append(ps.stem(w))

print(stem_words)
```

```
['mr.', 'smith', 'feel', 'relax', 'today', 'weather', 'usa', 'awesom', 'someth', 'troubl', 'u.s.a.', 'bird', 'fli']
```

Lemmatization reduces words to their base word, which is linguistically correct lemmas.

```
# Lemmetization
from nltk.stem.wordnet import WordNetLemmatizer
lem = WordNetLemmatizer()

lem_words=[]

for w in filtered_tokens:
     lem_words.append(lem.lemmatize(w, "v"))

print(lem_words)
```

```
['mr.', 'smith', 'feel', 'relax', 'today', 'weather', 'usa', 'awesome', 'something', 'trouble', 'u.s.a.', 'bird', 'fly']
```

```
d = {'Original': filtered_tokens, 'Stemming': stem_words, 'Lemmetization': lem_words}
df = pd.DataFrame(d)
df
```

	Original	Stemming	Lemmetization
0	mr.	mr.	mr.
1	smith	smith	smith
2	feeling	feel	feel
3	relaxed	relax	relax
4	today	today	today
5	weather	weather	weather
6	usa	usa	usa
7	awesome	awesom	awesome
8	something	someth	something
9	troubled	troubl	trouble
10	u.s.a.	u.s.a.	u.s.a.
11	birds	bird	bird
12	flying	fli	fly

Next steps: Generate code with df View recommended plots

```
Start coding or generate with AI.
```