Installation and Resource Downloads

```
!pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.11/dist-packages (3.9.1)
Requirement already satisfied: click in /usr/local/lib/python3.11/dist-packages (from nltk) (8.2.1)
Requirement already satisfied: joblib in /usr/local/lib/python3.11/dist-packages (from nltk) (1.5.1)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.11/dist-packages (from nltk) (2024.11.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from nltk) (4.67.1)
```

```
import nltk
nltk.download('punkt')       # For tokenization
nltk.download('averaged_perceptron_tagger')  # For POS tagging
nltk.download('stopwords')   # For stopwords
nltk.download('wordnet')     # For lemmatization
```

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

Tokenization

```
from nltk.tokenize import word_tokenize
import nltk

# Download the punkt resource if not already present
try:
    nltk.data.find('tokenizers/punkt/english.pickle')
except nltk.downloader.DownloadError:
    nltk.download('punkt')
```

```
text = "NLTK is a leading platform for building Python programs to work with human language data."
tokens = word_tokenize(text)
print(tokens)
```

```
---------------------------------------------------------------------------
LookupError                               Traceback (most recent call last)
/tmp/ipython-input-7-2011016556.py in <cell line: 0>()
     10
     11 text = "NLTK is a leading platform for building Python programs to work with human language data."
---> 12 tokens = word_tokenize(text)
     13 print(tokens)

                              5 frames
/usr/local/lib/python3.11/dist-packages/nltk/data.py in find(resource_name, paths)
    577     sep = "*" * 70
    578     resource_not_found = f"\n{sep}\n{msg}\n{sep}\n"
--> 579     raise LookupError(resource_not_found)
    580
    581

LookupError:
**********************************************************************
  Resource punkt_tab not found.
  Please use the NLTK Downloader to obtain the resource:

  >>> import nltk
  >>> nltk.download('punkt_tab')

  For more information see: https://www.nltk.org/data.html

  Attempted to load tokenizers/punkt_tab/english/

  Searched in:
    - '/root/nltk_data'
    - '/usr/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/local/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/local/lib/nltk_data'
**********************************************************************
```

Next steps: Explain error

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

# Download the punkt resource if not already present
try:
    nltk.data.find('tokenizers/punkt/english.pickle')
except Exception:
    nltk.download('punkt')

# Attempt to download punkt_tab as suggested by the error
try:
    nltk.data.find('tokenizers/punkt_tab/english/punkt_tab.p') # Adjusted path based on common resource naming
except Exception:
    nltk.download('punkt_tab')


text = "NLTK is a leading platform for building Python programs to work with human language data."

# Tokenize into sentences first
sentences = sent_tokenize(text)

# Tokenize each sentence into words
tokens = [word_tokenize(sent) for sent in sentences]

print(tokens)
```

```
[['NLTK', 'is', 'a', 'leading', 'platform', 'for', 'building', 'Python', 'programs', 'to', 'work', 'with', 'human', 'language', 'data',
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
```

Explanation: Splits the text into words and punctuation marks (tokens), supporting downstream text analysis tasks.

Sentence Segmentation

```
from nltk.tokenize import sent_tokenize

text = "NLP enables machines to use language. It is a rapidly evolving field."
sentences = sent_tokenize(text)
print(sentences)
```

```
['NLP enables machines to use language.', 'It is a rapidly evolving field.']
```

Explanation: Breaks the text into individual sentences for processing and analysis.

Part-of-Speech (POS) Tagging

```
import nltk

tokens = word_tokenize("NLTK processes text efficiently.")
pos_tags = nltk.pos_tag(tokens)
print(pos_tags)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
[('NLTK', 'NNP'), ('processes', 'VBZ'), ('text', 'RB'), ('efficiently', 'RB'), ('.', '.')]
```

Explanation: Assigns grammatical categories (e.g., noun, verb) to each token.

Lemmatization

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
tokens = word_tokenize("The leaves are falling.")

lemmas = [lemmatizer.lemmatize(token) for token in tokens]
print(lemmas)
```

```
['The', 'leaf', 'are', 'falling', '.']
```

Explanation: Converts words to their base form (e.g., "leaves" → "leaf", "falling" → "fall").

Named Entity Recognition (NER)

Gemini

```
import nltk
from nltk.tokenize import word_tokenize

# Download the words corpus if not already present
try:
    nltk.data.find('corpora/words')
except Exception:
    nltk.download('words')

# Download the maxent_ne_chunker resource if not already present (previous fixes)
try:
    nltk.data.find('chunkers/maxent_ne_chunker/english.pickle')
except Exception:
    nltk.download('maxent_ne_chunker')

# Download the maxent_ne_chunker_tab resource if not already present (previous fixes attempt)
try:
    nltk.data.find('chunkers/maxent_ne_chunker_tab/english_ace_multiclass.pickle')
except Exception:
    nltk.download('maxent_ne_chunker_tab')


sentence = "Apple is looking at buying U.K. startup for $1 billion."
tokens = word_tokenize(sentence)
pos_tags = nltk.pos_tag(tokens)
ner_tree = nltk.ne_chunk(pos_tags)
```

```
print(ner_tree)
```

```
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Unzipping corpora/words.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 maxent_ne_chunker_tab to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package maxent_ne_chunker_tab is already up-to-date!
(S
  (GPE Apple/NNP)
  is/VBZ
  looking/VBG
  at/IN
  buying/VBG
  U.K./NNP
  startup/NN
  for/IN
  $/$
  1/CD
  billion/CD
  ./.)
```

Explanation: Identifies persons, organizations, locations, and other entities in the text.

Stop Word Detection

```
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))
tokens = word_tokenize("This is a simple test.")

filtered = [w for w in tokens if w.lower() not in stop_words]
print(filtered)
```

```
['simple', 'test', '.']
```

Explanation: Removes common, low-meaning words (e.g., "is", "a", "the") to focus analysis on meaningful content.

Stemming

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()
tokens = word_tokenize("The leaves are falling.")

stems = [stemmer.stem(token) for token in tokens]
print(stems)
```

```
['the', 'leav', 'are', 'fall', '.']
```

Explanation: Reduces words to their root form, though not always a valid dictionary term (e.g., "falling" → "fall", "leaves" → "leav").

Dependency Parsing NLTK does not provide built-in dependency parsing like spaCy. For constituent parsing:

```
from nltk import CFG, ChartParser

# Example: parses require grammar and simple sentences
grammar = CFG.fromstring("""
S -> NP VP
NP -> DT NN
VP -> VB NP
DT -> 'the'
NN -> 'dog'
VB -> 'chased'
""")
```