**1. PREPROCESS TEXTUAL DATA: TOKENIZATION, LEMMATIZATION, STOPWORD REMOVAL**

**AIM :** Implement a preprocess textual data: Tokenization, lemmatization and stopword removal in natural language processing by using python.

**DESCRIPTION :**

1. **TOKENIZATION**

Tokenization is the process of splitting text into **smaller units (tokens)** such as words, sentences, or subwords.

* Makes text manageable and analyzable
* Tokens serve as the base input for most NLP models

**TYPES OF TOKENIZATION**

| **TYPE** | **DESCRIPTION** | **EXAMPLE** |
| --- | --- | --- |
| Word Tokenization | Splits text into individual words | "NLP is fun" → ["NLP", "is", "fun"] |
| Sentence Tokenization | Splits text into sentences | "Hi! How are you?" → ["Hi!", "How are you?"] |
| Subword Tokenization | Splits into meaningful sub-parts (used in transformers) | "playing" → ["play", "##ing"] (BERT-style) |

**Tools:**

* NLTK: word\_tokenize()
* SpaCy: nlp(text)
* Hugging Face Tokenizers

1. **LEMMATIZATION**

Lemmatization reduces words to their **base or dictionary form (lemma)**, considering the **context and part of speech**.

* Normalize words to a common form
* Helps in reducing sparsity in text analysis

**Example:**

| **Word** | **Lemma (Verb)** | **Lemma (Noun)** |
| --- | --- | --- |
| running | run | — |
| better | good | — |
| studies | study | — |

E.g., "He studies hard" → "He study hard"

**Tools:**

* **SpaCy**: token.lemma\_
* **NLTK**: WordNetLemmatizer

**LEMMATIZATION VS. STEMMING**

| **FEATURE** | **LEMMATIZATION** | **STEMMING** |
| --- | --- | --- |
| Linguistic Meaning | Preserves | May be lost |
| Output Word | Dictionary word | May not be real word |
| Accuracy | Higher | Lower |

1. **STOPWORD REMOVAL**

Stopwords are **common words** (e.g., “is”, “the”, “and”) that do not carry significant meaning and are **often removed** from text.

* Reduces noise
* Improves model focus on informative words

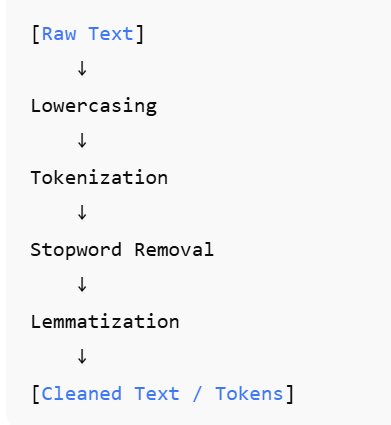
**Note:**

Removing stopwords is **task-dependent**; in sentiment analysis, words like "not" or "never" are important and **should be kept**.

**Tools:**

* NLTK: stopwords.words('english')
* SpaCy: token.is\_stop
* Custom stopword lists

**Typical Preprocessing Pipeline**



**Example:**

Input:

"Cats are running faster than dogs."

Steps:

1. **Lowercase**: "cats are running faster than dogs."
2. **Tokenize**: ["cats", "are", "running", "faster", "than", "dogs"]
3. **Remove Stopwords**: ["cats", "running", "faster", "dogs"]
4. **Lemmatize**: ["cat", "run", "fast", "dog"]

| **STEP** | **PURPOSE** | **EXAMPLE INPUT** | **EXAMPLE OUTPUT** |
| --- | --- | --- | --- |
| Tokenization | Split text into smaller units | "I love NLP" | ["I", "love", "NLP"] |
| Stopword Removal | Remove common filler words | ["I", "love", "NLP"] | ["love", "NLP"] |
| Lemmatization | Normalize words to base form | "loved", "dogs" | "love", "dog" |

**INSTALL REQUIRED LIBRARIES**

pip install nltk spacy pandas

python -m nltk.downloader punkt stopwords

python -m spacy download en\_core\_web\_sm import nltk

**PROGRAM**

import nltk

import spacy

import pandas as pd

from nltk.tokenize import word\_tokenize

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer

# Download required resources

nltk.download('punkt')

nltk.download('stopwords')

# Load models and tools

stop\_words = set(stopwords.words('english'))

stemmer = PorterStemmer()

nlp = spacy.load("en\_core\_web\_sm")

def preprocess\_text\_to\_table(sentence):

# Tokenize and lowercase

tokens = word\_tokenize(sentence.lower())

data = []

for token in tokens:

if token.isalpha(): # Filter out punctuation/numbers

is\_stop = token in stop\_words

lemma = nlp(token)[0].lemma\_

stem = stemmer.stem(token)

data.append({

"Token": token,

"Is Stopword": is\_stop,

"Stem": stem,

"Lemma": lemma

})

# Create and return DataFrame

df = pd.DataFrame(data)

return df

# Example usage

if \_\_name\_\_ == "\_\_main\_\_":

sentence = "The quick brown foxes were jumping over the lazy dogs."

result\_df = preprocess\_text\_to\_table(sentence)

print("\nPreprocessed Table:\n")

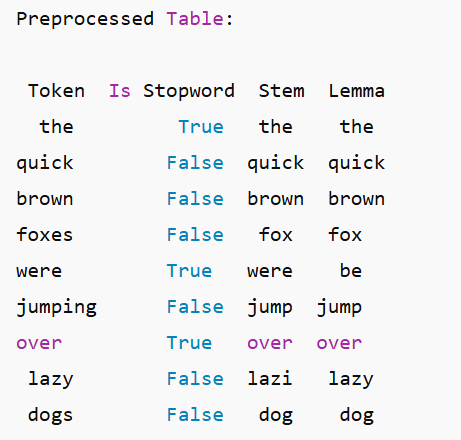
print(result\_df.to\_string(index=False))

**EXAMPLE OUTPUT**

For the input sentence:

"The quick brown foxes were jumping over the lazy dogs."

**OUTPUT**

****

**RESULT :**

Hence, the preprocess textual data: tokenization, lemmatization, stopword removal has been successfully implemented.