Department of Computer Science and Engineering PES University, Bangalore, India

Lecture Notes
Python for Computational Problem Solving UE25CS151A

Lecture 72

Natural Language Toolkit [NLTK]

By,
Prof. Kundhavai K R,
Assistant Professor
Dept. of CSE, PESU

Verified by,
PCPS Team -
2025

Introduction to NLTK:

Objective

Learn the basics of NLTK for text processing, including installation, tokenization, stop word removal, and word frequency analysis, with hands-on Python examples.

1. What is NLTK?

NLTK is a Python library for working with text data, perfect for beginners exploring Natural Language Processing (NLP). It helps with tasks like:

- Splitting text into words or sentences (tokenization).
- Filtering out common words (e.g., "the", "is").
- Counting word occurrences to find key themes.

Why Learn NLTK?

- Simplifies text analysis for projects like chatbots or text mining.
- Widely used in education and industry.

2. Installation and Setup Steps

Install NLTK using pip:

```
pip install nltk
```

Download required datasets in Python:

```
import nltk
#Punkt tokenizer model, which is used for tokenization.
nltk.download('punkt')
#Downloads a corpus of stop words for multiple languages.
nltk.download('stopwords')
#Downloads a tabular version of the Punkt tokenizer data.
nltk.download('punkt_tab')
print("NLTK is ready!")
```

Notes

- Run the pip command in your terminal or command prompt.
- The nltk.download commands fetch datasets needed for text processing.
- Verify setup by running the code (no errors means success).

3. Tokenization

Concept

Tokenization breaks text into smaller units called tokens (e.g., words or sentences). It's the first step in text analysis.

Types

- **Word Tokenization**: Splits text into words and punctuation.
- **Sentence Tokenization**: Splits text into sentences.

Example 1: Word Tokenization

```
from nltk.tokenize import word_tokenize
text = "Python is fun to learn!"
words = word_tokenize(text)
print("Words:", words)
```

Words: ['Python', 'is', 'fun', 'to', 'learn', '!']

Explanation:

- word_tokenize splits the text into a list of words and punctuation.

Example 2: Sentence Tokenization

```
from nltk.tokenize import sent_tokenize
text = "I love Python. NLTK is awesome!"
sentences = sent_tokenize(text)
print("Sentences:", sentences)
```

Sentences: ['I love Python.', 'NLTK is awesome!']

4. Stop Word Removal

Concept

Stop words are common words (e.g., "the", "and", "is") that often don't add meaning in text analysis. Removing them helps focus on important words.

Example

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
# Sample text
text = "The dog runs fast and jumps high."
# Load English stop words
stop_words = set(stopwords.words('english')) #loads a list of English stop words.
# Tokenize the text into words
words = word_tokenize(text)
# Create an empty list for filtered words
filtered_words = []
# Loop through each word and keep it if it's not a stop word
for word in words:
   if word.lower() not in stop_words:
      filtered_words.append(word)
# Print the filtered words
print("Filtered Words:", filtered_words)
```

Output:
Filtered Words: ['dog', 'runs', 'fast', 'jumps', 'high', '.']

5. Word Frequency Analysis

Concept

Word frequency counts how often each word appears in a text, helping identify key terms.

Example

```
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
text = "Python is fun. Python is easy. I love Python."
words = word_tokenize(text.lower())  # Case-insensitive
freq = FreqDist(words)
print("Top 3 Words:", freq.most_common(3))
```

Top 3 Words: [('python', 3), ('.', 3), ('is', 2)]

Explanation:

- FreqDist creates a dictionary of word counts.
- most_common(3) shows the top 3 most frequent words.

--END--

NLTK documentation: http://www.nltk.org/