Basic Word Tokenization using NLTK

Overview

This notebook demonstrates how to perform basic word tokenization on a sample text using the **Natural Language Toolkit (NLTK)** library in Python. Tokenization is the process of breaking text into smaller parts, such as words or sentences, which are known as "tokens." This technique is commonly used in natural language processing (NLP) to prepare text for analysis.

Steps Included

1. **Install NLTK**: Install the NLTK library to access text processing functions.
2. **Import Libraries**: Import the necessary functions from NLTK for tokenization and frequency distribution.
3. **Define a Sample Text**: Use a sample paragraph for demonstration, containing multiple sentences.
4. **Tokenize the Text**: Split the text into individual words (tokens).
5. **Display Tokens**: Print the resulting list of tokens.
6. **Count Tokens**: Count the total number of tokens.
7. **Frequency Distribution**: Use `FreqDist` to identify the frequency of each token.
8. **Display Most Common Tokens**: Identify and print the 10 most common tokens in the text.

Detailed Code Documentation

Step 1: Install NLTK

```
# NLTK is a popular library for natural language processing.

# Running this command installs the library in the Colab environment.

!pip install nltk
```

Step 2: Import Libraries

```
# Import the necessary functions from NLTK:

# - word_tokenize: This function splits a string of text into individual words.

# - FreqDist: This function generates a frequency distribution for the tokens,

#   which allows us to see how often each word appears in the text.

import nltk

from nltk.tokenize import word_tokenize
```

```
from nltk.probability import FreqDist
```

Step 3: Define the Text

```
# This variable 'text' contains a sample paragraph with at least four sentences.
# This text will be used to demonstrate tokenization and word frequency analysis.
text = "Machine learning is a fascinating field of study. It involves teaching computers to learn from data. With machine learning, computers can make predictions and decisions. The applications are endless, from self-driving cars to personalized recommendations."
```

Step 4: Tokenize the Text

```
# The word_tokenize function from NLTK is applied to 'text' to break it down into individual tokens (words and punctuation).
# This process converts the paragraph into a list of words, which we can then analyze further.
tokens = word_tokenize(text)
```

Step 5: Print the Tokens

```
# Display the list of tokens generated in the previous step. Each word and punctuation mark from the text appears as an element in this list.
print("Tokens:", tokens)
```

Step 6: Count the Number of Tokens

```
# Use the len() function to calculate the total number of tokens in the text.
# This provides a quick way to see how many individual words or symbols were extracted from the paragraph.
num_tokens = len(tokens)
print("Number of tokens:", num_tokens)
```

Step 7: Identify the Frequency of Each Token

```
# The FreqDist function is used to calculate the frequency distribution of the tokens.
# This function counts how often each word appears in the list of tokens, allowing us to analyze common and rare words.
```

```
freq_dist = FreqDist(tokens)
```

Step 8: Print the Frequency Distribution

```
# Display the frequency of each token in the text, showing how many times each word occurs.

print("Token Frequencies:", freq_dist)
```

Step 9: Display the Most Common Tokens

```
# Retrieve and print the 10 most common tokens and their frequencies.

# This step helps us identify the words that appear most frequently in the text.

print("Most Common Tokens:", freq_dist.most_common(10))
```

Explanation of Output

- **Tokens**: A list of words and punctuation marks extracted from the input text.
- **Number of Tokens**: The total count of tokens found in the text.
- **Token Frequencies**: A dictionary-like output showing each token and the number of times it appears in the text.
- **Most Common Tokens**: The top 10 most frequently occurring tokens in the text, along with their counts.