Veermata Jijabai Technological Institute, Mumbai 400019

Experiment No.: 01

Aim: Use of Natural Language Toolkit (NLTK): Computing with Language: Texts and Words, Tokenization, Segmentation, Texts as Lists of Words, Simple Statistic Generation

Name: Kiran K Patil

Enrolment No.: 211070904

Branch: Computer Engineering

Batch: D

Theory:

Natural Language Toolkit (NLTK):

Natural Language Toolkit (NLTK) is a comprehensive library in Python that is widely used for natural language processing (NLP) and text analysis. It provides tools, resources, and algorithms to work with human language data. NLTK was developed by researchers at the University of Pennsylvania and is a valuable resource for various NLP tasks. Here's an introduction to NLTK:

Key Features and Capabilities:

1. Text Processing: NLTK provides a range of text processing libraries for tasks like tokenization, stemming, lemmatization, and more.
2. Corpus and Language Resources: NLTK includes a wide variety of text corpora and lexical resources for different languages. These resources are valuable for research and experimentation.
3. Text Classification: It supports text classification tasks, including sentiment analysis, document categorization, and spam detection.
4. Parsing and Semantic Analysis: NLTK allows you to parse and analyze sentence and text structures, which is important for understanding the meaning of sentences.
5. Part-of-Speech Tagging: NLTK provides tools for part-of-speech tagging, which involves labeling words in a sentence with their corresponding parts of speech (e.g., noun, verb, adjective).
6. Named Entity Recognition: It can identify and classify named entities in text, such as names of people, organizations, dates, and more.
7. Concordance and Collocations: NLTK helps you find concordances (instances of a word appearing in context) and collocations (words that frequently appear together).
8. Frequency Distributions: NLTK's FreqDist class allows you to compute and visualize word frequencies in a text.
9. Machine Learning Integration: NLTK can be used in conjunction with machine learning libraries for tasks like text classification and sentiment analysis.

1. **Texts and Words:**
    - In natural language processing (NLP), text is often analyzed as a collection of words. NLTK provides tools for processing and analyzing text data.
    - A text can be considered as a sequence of words or tokens, and NLTK helps us work with these tokens for various NLP tasks.

2. **Tokenization:**

   - Tokenization is the process of breaking down a text into individual words or tokens.
   - NLTK's word_tokenize() function can split a given text into a list of tokens, making it easier to work with individual words.

3. **Segmentation:**

   - Segmentation involves splitting a text into meaningful segments, typically sentences or paragraphs.
   - NLTK's sent_tokenize() function allows you to segment a text into a list of sentences, which is useful for tasks like text summarization, translation, or sentiment analysis.

4. **Texts as Lists of Words:**

   - In NLP, text can be treated as a list of words or tokens, enabling various text analysis and processing operations.
   - NLTK facilitates the conversion of a text into a list of words, making it accessible for further manipulation.

5. **Simple Statistic Generation:**

   - NLTK provides tools for generating basic statistics from text data.
   - The FreqDist class allows you to calculate the frequency of each word in a text, which can be useful for tasks like identifying the most common words or building word clouds.
   - Additionally, we can calculate simple text statistics, such as lexical diversity, to gain insights into the variety of words used in a text.

In the experiment, we demonstrated how to use NLTK for these tasks. You can use NLTK for more advanced text analysis, including sentiment analysis, named entity recognition, part-of-speech tagging, and more, by building on these fundamental concepts and tools. NLTK is a valuable resource for working with natural language text data in the field of natural language processing and computational linguistics.

Implementation:

```
import nltk
nltk.download('punkt')
nltk.download('stopwords')

from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from collections import Counter
import os

# Define the input file path
input_file = 'sample.txt'

# Read the input text from the file
with open(input_file, 'r', encoding='utf-8') as file:
    text = file.read()

# Task 3: Tokenize sentences and words
sentences = sent_tokenize(text)
words_in_sentences = [word_tokenize(sentence) for sentence in
sentences]

# Task 4: Create a file containing word frequency
word_freq = Counter([word.lower() for sentence in words_in_sentences
for word in sentence])
with open('output/word_frequency.txt', 'w', encoding='utf-8') as file:
    for word, freq in word_freq.items():
        file.write(f'{word}: {freq}\n')

# Task 5: Remove stopwords
stop_words = set(stopwords.words('english'))
filtered_words = [word for sentence in words_in_sentences for word in
sentence if word.lower() not in stop_words]

# Task 6: Create a vocabulary file
unique_words = set(filtered_words)
with open('output/vocabulary.txt', 'w', encoding='utf-8') as file:
    for word in unique_words:
        file.write(f'{word}\n')

# Task 7: Create index-to-word and word-to-index files
index_to_word = {i: word for i, word in enumerate(unique_words)}
word_to_index = {word: i for i, word in enumerate(unique_words)}

with open('output/index_to_word.txt', 'w', encoding='utf-8') as file:
    for index, word in index_to_word.items():
        file.write(f'{index}: {word}\n')
```

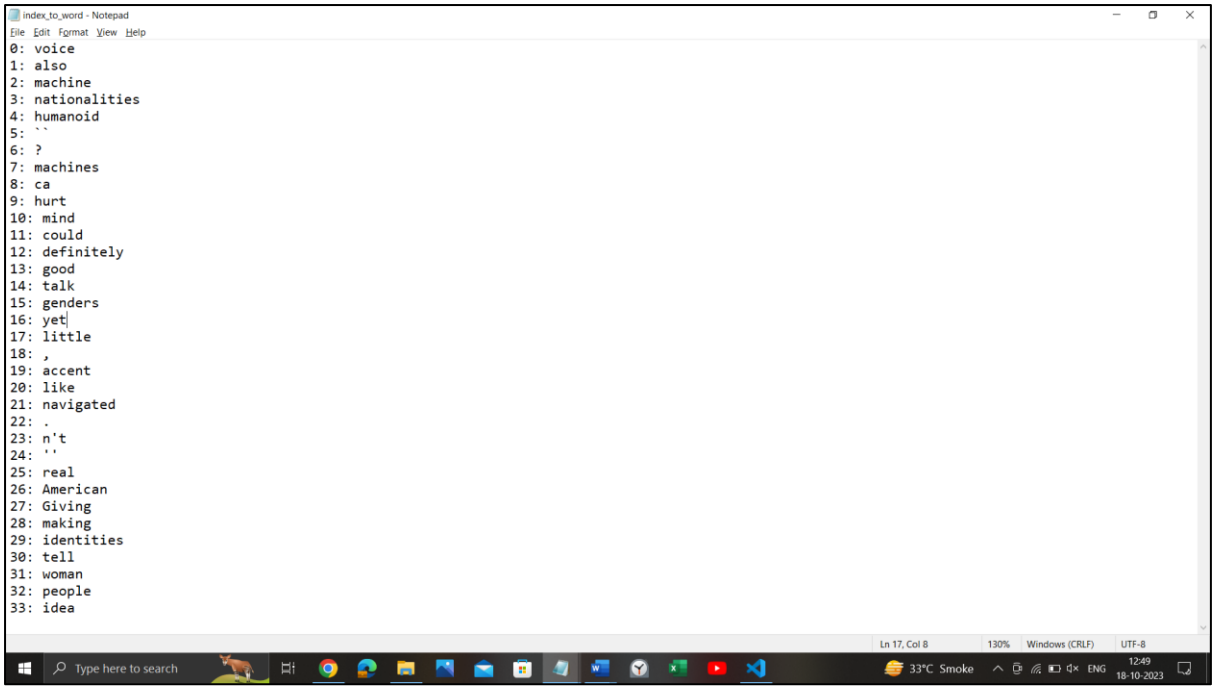
```
with open('output/word_to_index.txt', 'w', encoding='utf-8') as file:
    for word, index in word_to_index.items():
        file.write(f'{word}: {index}\n')

print("Tasks completed successfully!")
```

Sample data :

```
"""The voice that navigated was definitely that of a machine, and yet you could tell that the machine was a woman, which hurt my mind a little. How can machines have genders? The machine also had an American accent. How can machines have nationalities? This can't be a good idea, making machines talk like real people, can it? Giving machines humanoid identities?"""
```

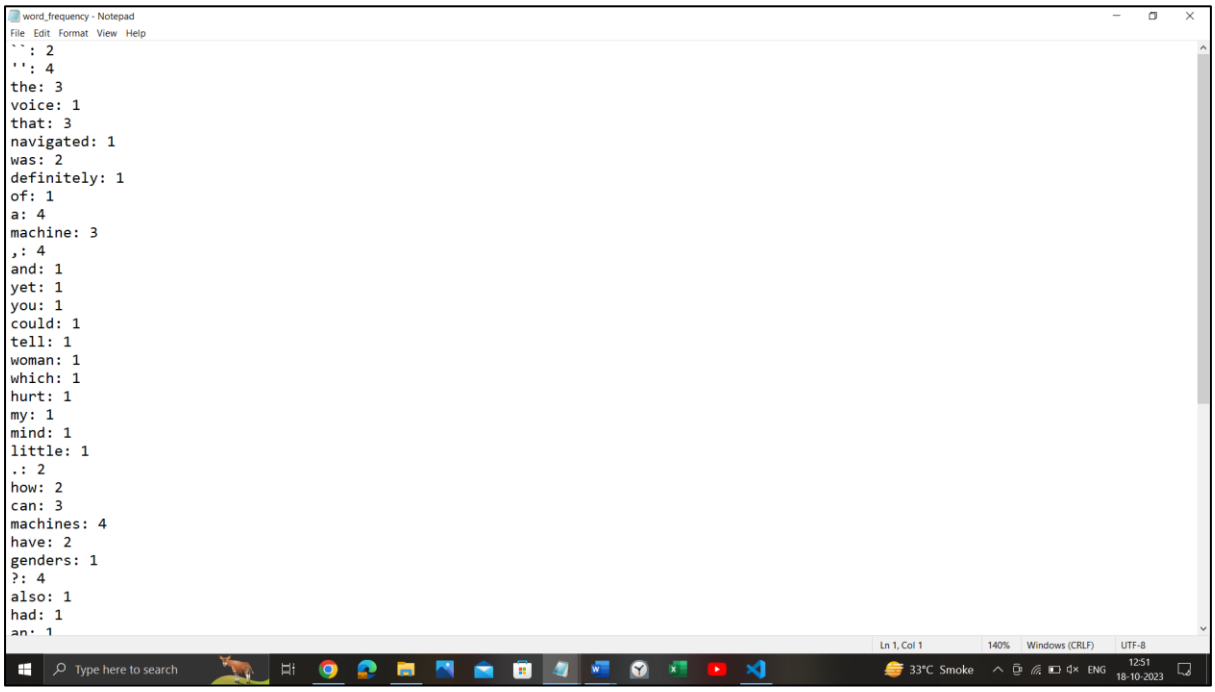
Output :

1.Index_to_word

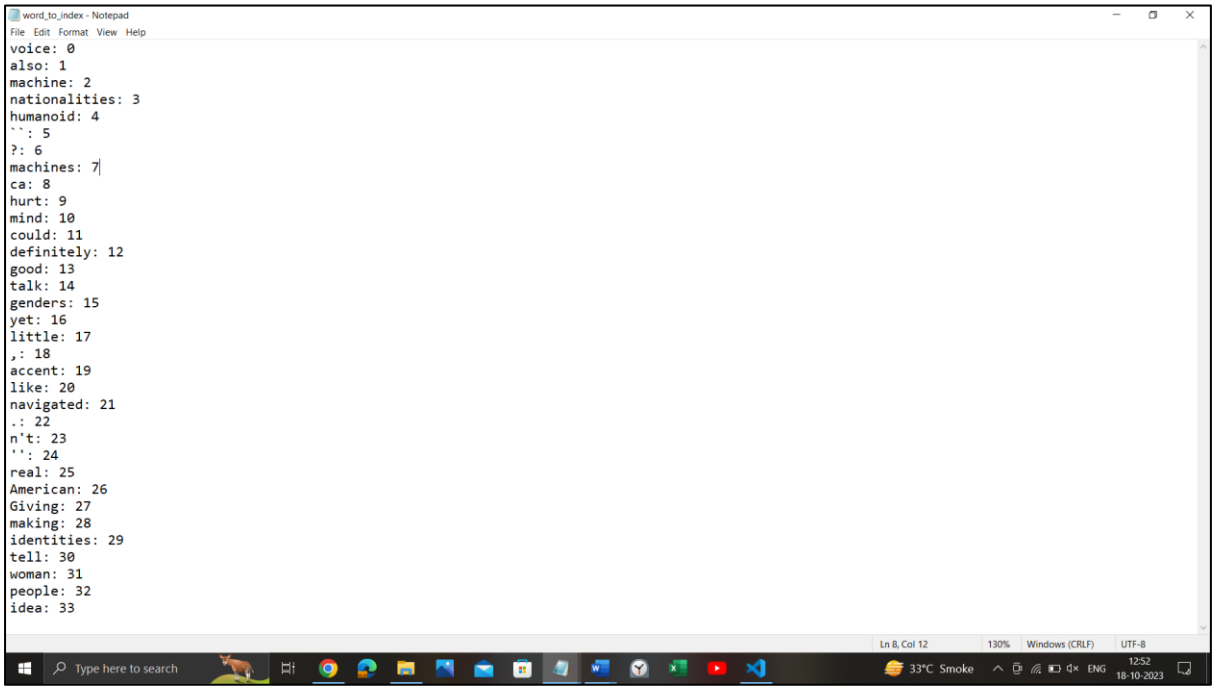


2.Vocablary :

3. Word frequency



4.Word to index:



Conclusion:

In this experiment, we introduced the Natural Language Toolkit (NLTK) and explored its fundamental capabilities for text processing and natural language processing (NLP). We covered tokenization, segmentation, and basic text statistics, setting the foundation for more advanced NLP tasks. NLTK is a powerful Python library that is widely used for working with human language data, making it a valuable resource for researchers, developers, and data scientists in the field of NLP.