SATHYABAMA INSTITUTE OF SCIENCE & TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCSA 2604 NATURAL LANGUAGE PROCESSING LAB

LAB 2: WORD GENERATION

AIM: Word generation using NLTK

PROCEDURE:

Loading Resources: Install NLTK if necessary and download the 'punkt' and 'gutenberg' resources to tokenize text and access the Gutenberg corpus.

Loading Corpus: Load a specific corpus (here, the Gutenberg corpus) from NLTK, which provides a collection of words.

Bigram Generation: Create bigrams (pairs of consecutive words) from the loaded corpus to understand word associations and probabilities.

Text Generation Loop: Generate text by selecting subsequent words based on the last word generated, using the bigram model. This process is repeated a specified number of times (20 iterations in this case).

Random Selection: Randomly choose the next word from the possibilities obtained from the bigrams that follow the last generated word.

Display Generated Text: Print the generated sequence of words.

The following algorithm outlines the steps involved in generating text based on a chosen corpus using bigram models and random selection of subsequent words for text generation.

ALGORITHM:

1. Install and Import Libraries: Install NLTK (!pip install nltk) and import the required libraries (nltk and random).
2. Download NLTK Resources: Download NLTK resources, specifically the 'punkt' and 'gutenberg' corpora.
3. Load Corpus: Load a corpus from NLTK (e.g., Gutenberg corpus).
4. Create Bigram Model: Generate a list of bigrams (pairs of consecutive words) from the loaded corpus.
5. Choose Starting Word: Select a starting word for text generation.
6. Generate Text:

    a. Iterate a specified number of times (here, 20 iterations).
    b. For each iteration:
    c. Find all possible words that follow the last generated word in the bigrams.
    d. Randomly select a word from the possible words to continue the sequence.
    e. Append the selected word to the generated text.
7. Output Generated Text: Display the generated text.

PROGRAM:

```
!pip install nltk

import nltk

import random


# Download NLTK resources (run only once if not downloaded)

nltk.download('punkt')

nltk.download('gutenberg')


# Load a corpus (for example, the Gutenberg corpus)

words = nltk.corpus.gutenberg.words()


# Create a bigram model

bigrams = list(nltk.bigrams(words))


# Choose a starting word (you can choose any word from the corpus)

starting_word = "the"


generated_text = [starting_word]


# Generate 20 words of text

for _ in range(20):

    # Get all bigrams that start with the last generated word
    possible_words = [word2 for (word1, word2) in bigrams if word1.lower() ==
generated_text[-1].lower()]

    # Choose a word randomly from the possible options
    next_word = random.choice(possible_words)
    generated_text.append(next_word)

# Print the generated text
print(' '.join(generated_text))
```

OUTPUT:

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk)
(8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk)
(1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages
(from nltk) (2023.6.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk)
(4.66.1)
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package gutenberg to /root/nltk_data...
[nltk_data]   Package gutenberg is already up-to-date!
the sea , avoided . Hence it my life in the left to keep my witness shall no hesitation , accept
```