Experiment No.3
Apply various other text preprocessing techniques for any given text: Stop Word Removal, Lemmatization / Stemming.
Date of Performance:
Date of Submission:

Aim: Apply various other text preprocessing techniques for any given text: Stop Word Removal, Lemmatization / Stemming.

Objective: To write a program for Stop word removal from a sentence given in English and any Indian Language.

Theory:

The process of converting data to something a computer can understand is referred to as preprocessing. One of the major forms of pre-processing is to filter out useless data. In natural language processing, useless words (data), are referred to as stop words.

Stopwords are the most common words in any natural language. For the purpose of analyzing text data and building NLP models, these stopwords might not add much value to the meaning of the document.

Stop Words: A stop word is a commonly used word (such as "the", "a", "an", "in") that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. We need to perform tokenization before removing any stopwords.

Why do we need to Remove Stopwords?

Removing stopwords is not a hard and fast rule in NLP. It depends upon the task that we are working on. For tasks like text classification, where the text is to be classified into different categories, stopwords are removed or excluded from the given text so that more focus can be given to those words which define the meaning of the text.

Here are a few key benefits of removing stopwords:

- On removing stopwords, dataset size decreases and the time to train the model also decreases
- Removing stopwords can potentially help improve the performance as there are fewer and only meaningful tokens left. Thus, it could increase classification accuracy

- Even search engines like Google remove stopwords for fast and relevant retrieval of data from the database

We can remove stopwords while performing the following tasks:

- Text Classification ○ Spam Filtering ○ Language Classification ○ Genre Classification
- Caption Generation ● Auto-Tag Generation

Avoid Stopword Removal

- Machine Translation
- Language Modeling
- Text Summarization ● Question-Answering problems

Different Methods to Remove Stopwords

1. **Stopword Removal using NLTK**

   NLTK, or the Natural Language Toolkit, is a treasure trove of a library for text preprocessing. It's one of my favorite Python libraries. NLTK has a list of stopwords stored in 16 different languages.

   You can use the below code to see the list of stopwords in NLTK:

   ```
   import nltk
   from nltk.corpus import stopwords set(stopwords.words('english'))
   ```

2. **Stopword Removal using spaCy:**

   spaCy is one of the most versatile and widely used libraries in NLP. We can quickly and efficiently remove stopwords from the given text using SpaCy.

   It has a list of its own stopwords that can be imported as **STOP_WORDS** from the **spacy.lang.en.stop_words** class.

3. **Stopword Removal using Gensim**

Gensim is a pretty handy library to work with on NLP tasks. While pre-processing, gensim provides methods to remove stopwords as well. We can easily import the remove_stopwords method from the class gensim.parsing.preprocessing.

Implementation:

```
!pip install nltk

text = '"I have 2 cats. They eat 3 times a day."'

text
```

```
'I have 2 cats. They eat 3 times a day.'
```

```
from nltk.corpus import stopwords

stop_words = stopwords.words('english')

from nltk.tokenize import word_tokenize
words = word_tokenize(text)

holder = list()
for w in words:
    if w not in set(stop_words):
        holder.append(w)

holder
```

```
['I', '2', 'cats', '.', 'They', 'eat', '3', 'times', 'day', '.']
```

```
holder = [w for w in words if w not in set(stop_words)]
print(holder)
```

```
['I', '2', 'cats', '.', 'They', 'eat', '3', 'times', 'day', '.']
```

```
from nltk.stem import PorterStemmer, SnowballStemmer, LancasterStemmer

porter = PorterStemmer()
snow = SnowballStemmer(language = 'english')
lancaster = LancasterStemmer()

words = ['play', 'plays', 'played', 'playing', 'player']
porter_stemmed = list()
```

```
for w in words:
    stemmed_words = porter.stem(w)
    porter_stemmed.append(stemmed_words)

porter_stemmed
['play', 'play', 'play', 'play', 'player']
snow_stemmed = list()
for w in words:
    stemmed_words = snow.stem(w)
    snow_stemmed.append(stemmed_words)

snow_stemmed
['play', 'play', 'play', 'play', 'player']
snow_stemmed = [snow.stem(x) for x in words]
print (snow_stemmed)
['play', 'play', 'play', 'play', 'player']
lancaster_stemmed = list()
for w in words:
    stemmed_words = lancaster.stem(w)
    lancaster_stemmed.append(stemmed_words)

lancaster_stemmed
['play', 'play', 'play', 'play', 'play']
lancaster_stemmed = [lancaster.stem(x) for x in words]
print (lancaster_stemmed)
['play', 'play', 'play', 'play', 'play']
from nltk.stem import WordNetLemmatizer
wordnet = WordNetLemmatizer()

lemmatized = [wordnet.lemmatize(x) for x in words]

lemmatized
['play', 'play', 'played', 'playing', 'player']
```

Conclusion:

Comment on the tools used for stop word removal of Indian language input and explain the steps involved.

Stop word removal is an important preprocessing step in Natural Language Processing (NLP) to filter out commonly occurring words that don't contribute significantly to the meaning of a sentence (e.g., "is," "the"). For Indian languages, stop word removal requires special tools and techniques due to the diversity and complexity of these languages.

Tools for Stop Word Removal in Indian Languages:

1. NLTK (Natural Language Toolkit): While primarily developed for English, NLTK can be extended for Indian languages by adding custom stop word lists.
2. Indic NLP Library: This is a specialized library for Indian languages that includes stop word lists and other NLP functionalities.
3. Spacy with custom stop word lists: Spacy allows you to create custom language models, where you can include stop words for specific Indian languages.
4. Polyglot: It provides NLP functionality for multiple languages, including several Indian languages.
5. Snowball Stemmer: While used mainly for stemming, it supports stop word removal through rule-based mechanisms in Indian languages.
6. Pre-trained models from Hugging Face: Some models support Indian languages, offering in-built stop word lists.