Experiment No.3
Apply Stop Word Removal on given English and Indian Language Text
Date of Performance:
Date of Submission:

Aim: Apply Stop Word Removal on given English and Indian Language Text.

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 pre-processing. 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 stop words 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.

Output:

Library required

```
In [ ]: !pip install nltk

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)
```

Text

```
In [ ]: text = 'The general trend in IR systems over time has been from standard use of quite large stop lists (200-300 terms) to v

In [ ]: text

Out[ ]: 'TON 618 is a hyperluminous, broad-absorption-line, radio-loud quasar and Lyman-alpha blob located near the border of the co
nstellations Canes Venatici and Coma Berenices, with the projected comoving distance of approximately 18.2 billion light-yea
rs from Earth.'
```

Stopwords

In [ ]:
```
from nltk.corpus import stopwords
```

In [ ]:
```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[ ]: True

In [ ]:
```
stop_words = stopwords.words('english')
```

In [ ]:
```
from nltk.tokenize import word_tokenize
words = word_tokenize(text)
```

Applying stop words

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

In [ ]:
```
holder
```

Out[ ]:
```
['The',
 'general',
 'trend',
 'IR',
 'systems',
 'time',
 'standard',
 'use',
 'quite',
 'large',
 'stop',
 'lists',
 '(',
 '200-300',
 'terms',
 ')',
 'small',
 'stop',
 'lists',
 '(',
 '7-12',
 'terms',
 ')',
 'stop',
 'list',
```

List Comprehension for stop words

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

```
['The', 'general', 'trend', 'IR', 'systems', 'time', 'standard', 'use', 'quite', 'large', 'stop', 'lists', '(', '200-300', 'terms', ')', 'small', 'stop', 'lists', '(', '7-12', 'terms', ')', 'stop', 'list', 'whatsoever', '.', 'Web', 'search', 'engines', 'generally', 'use', 'stop', 'lists', '.']
```

Stemming

In [ ]:
```
from nltk.stem import PorterStemmer, SnowballStemmer, LancasterStemmer
```

In [ ]:
```
porter = PorterStemmer()
snow = SnowballStemmer(language = 'english')
lancaster = LancasterStemmer()
```

In [ ]:
```
words = ['play', 'plays', 'played', 'playing', 'player']
```

Porter Stemmer

In [ ]:
```
porter_stemmed = list()
for w in words:
    stemmed_words = porter.stem(w)
    porter_stemmed.append(stemmed_words)
```

In [ ]:
```
porter_stemmed
```

Out[ ]:
```
['play', 'play', 'play', 'play', 'player']
```

Porter Stemmer List Comprehension

In [ ]:
```
porter_stemmed = [porter.stem(x) for x in words]
print (porter_stemmed)
```

```
['play', 'play', 'play', 'play', 'player']
```

Snowball Stemmer

In [ ]:
```
snow_stemmed = list()
for w in words:
    stemmed_words = snow.stem(w)
    snow_stemmed.append(stemmed_words)
```

In [ ]:
```
snow_stemmed
```

Out[ ]:
```
['play', 'play', 'play', 'play', 'player']
```

Snowball Stemmer List Comprehension

In [ ]:
```
snow_stemmed = [snow.stem(x) for x in words]
print (snow_stemmed)
```

```
['play', 'play', 'play', 'play', 'player']
```

Lancaster Stemmer

In [ ]:
```
lancaster_stemmed = list()
for w in words:
    stemmed_words = lancaster.stem(w)
    lancaster_stemmed.append(stemmed_words)
```

Lancaster Stemmer

In [ ]:
```
lancaster_stemmed = list()
for w in words:
    stemmed_words = lancaster.stem(w)
    lancaster_stemmed.append(stemmed_words)
```

In [ ]:
```
lancaster_stemmed
```

Out[ ]: ['play', 'play', 'play', 'play', 'play']

Lancaster Stemmer List Comprehension

In [ ]:
```
lancaster_stemmed = [lancaster.stem(x) for x in words]
print (lancaster_stemmed)
```

['play', 'play', 'play', 'play', 'play']

Lemmatization : This has a more expansive vocabulary than Stemming

In [ ]:
```
from nltk.stem import WordNetLemmatizer
wordnet = WordNetLemmatizer()
```

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

In [ ]:
```
lemmatized
```

Out[ ]: ['play', 'play', 'played', 'playing', 'player']

Conclusion:

There are a number of tools available for stop word removal of Indian language input. Some of the most popular tools include:

iNLTK: iNLTK is a Python library for natural language processing (NLP) in Indian languages. It includes a stop word list for a variety of Indian languages.

Mila NMT: Mila NMT is a machine translation toolkit that includes a stop word list for Indian languages.

Indic NLP Library: The Indic NLP Library is a Python library for NLP in Indian languages. It includes a stop word list for a variety of Indian languages.

spaCy: spaCy is a Python library for NLP. It includes a stop word list for Indian languages, but it is not as comprehensive as the other tools listed above.