Vidyavardhini's College of Engineering & Technology

Department of Computer Science and Engineering (Data Science)

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 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:**

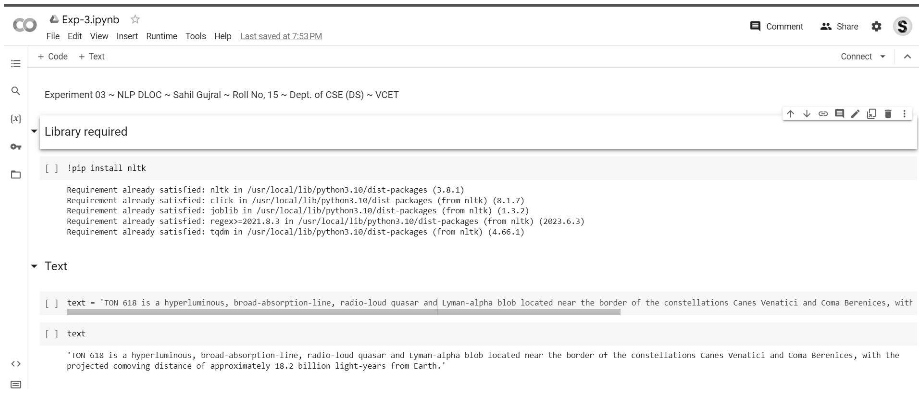
   spaCy is one of the most versatile and widely used libraries in NLP. We can quickly and efficiently remove stopwords from the given text using SpaCy.

   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:



Exp-3.ipynb
File Edit View Insert Runtime Tools Help Last saved at 7:53 PM
Comment Share
+ Code + Text
Connect

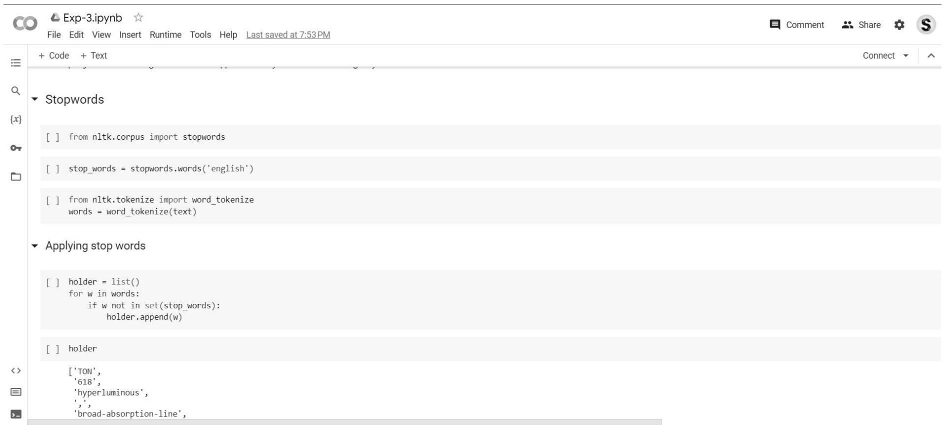
Experiment 03 ~ NLP DLOC ~ Sahil Gujral ~ Roll No, 15 ~ Dept. of CSE (DS) ~ VCET

Library required

```
[ ] !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

```
[ ] text = 'TON 618 is a hyperluminous, broad-absorption-line, radio-loud quasar and Lyman-alpha blob located near the border of the constellations Canes Venatici and Coma Berenices, witł

[ ] text

    'TON 618 is a hyperluminous, broad-absorption-line, radio-loud quasar and Lyman-alpha blob located near the border of the constellations Canes Venatici and Coma Berenices, with the
    projected comoving distance of approximately 18.2 billion light-years from Earth.'
```



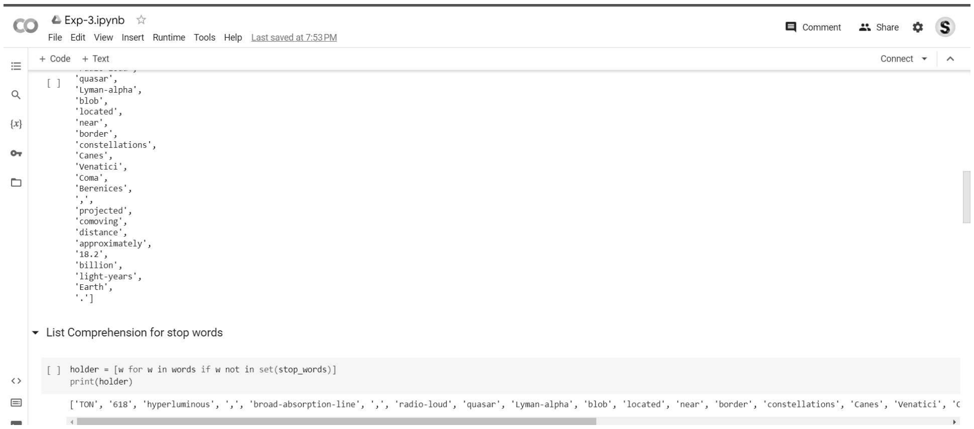
Exp-3.ipynb
File Edit View Insert Runtime Tools Help Last saved at 7:53 PM
Comment Share
+ Code + Text
Connect

Stopwords

```
[ ] from nltk.corpus import stopwords

[ ] stop_words = stopwords.words('english')

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

Applying stop words

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

[ ] holder

    ['TON',
     '618',
     'hyperluminous',
     ',',
     'broad-absorption-line',
```

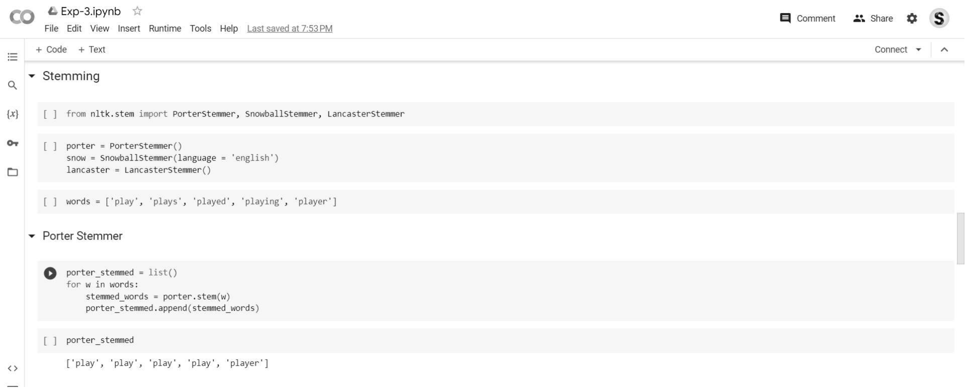

Exp-3.ipynb
File Edit View Insert Runtime Tools Help Last saved at 7:53 PM
Comment Share
+ Code + Text
Connect

```
 'quasar',
 'Lyman-alpha',
 'blob',
 'located',
 'near',
 'border',
 'constellations',
 'Canes',
 'Venatici',
 'Coma',
 'Berenices',
 ',',
 'projected',
 'comoving',
 'distance',
 'approximately',
 '18.2',
 'billion',
 'light-years',
 'Earth',
 '.']
```

List Comprehension for stop words

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

```
['TON', '618', 'hyperluminous', ',', 'broad-absorption-line', ',', 'radio-loud', 'quasar', 'Lyman-alpha', 'blob', 'located', 'near', 'border', 'constellations', 'Canes', 'Venatici', 'C
```


Exp-3.ipynb
File Edit View Insert Runtime Tools Help Last saved at 7:53 PM
Comment Share
+ Code + Text
Connect

Stemming

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

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

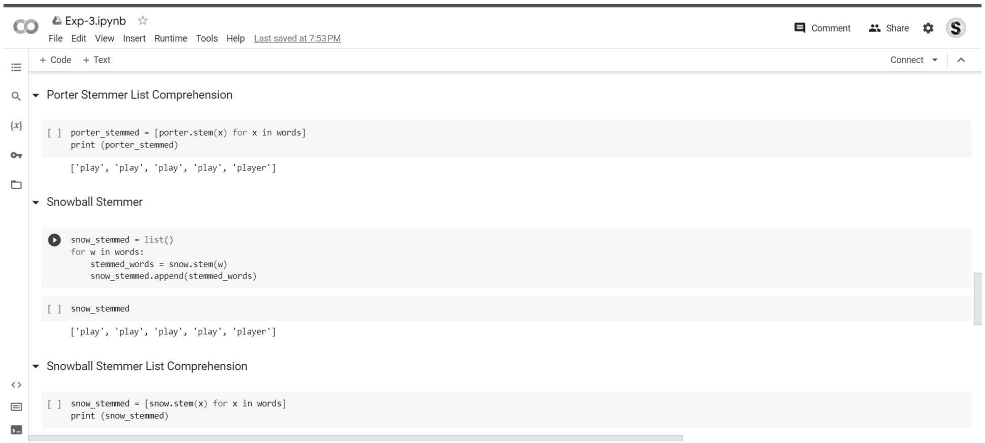
```
words = ['play', 'plays', 'played', 'playing', 'player']
```

Porter Stemmer

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

```
porter_stemmed
```

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


Exp-3.ipynb
File Edit View Insert Runtime Tools Help Last saved at 7:53 PM
Comment Share
+ Code + Text
Connect

Porter Stemmer List Comprehension

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

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

Snowball Stemmer

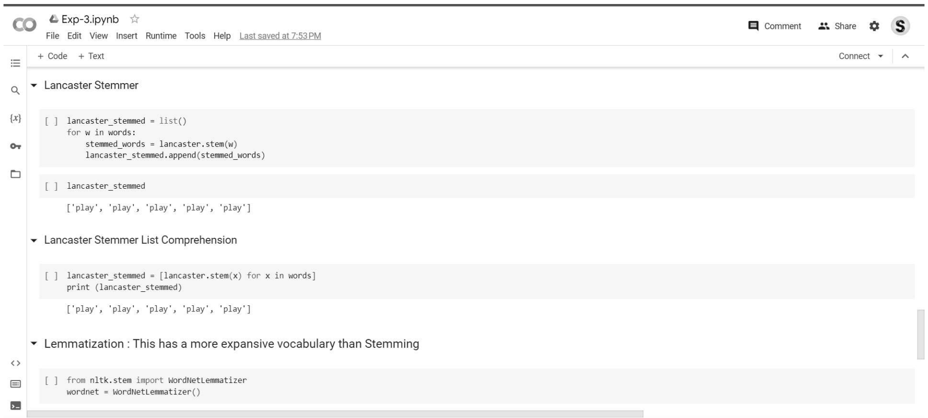
```
snow_stemmed = list()
for w in words:
    stemmed_words = snow.stem(w)
    snow_stemmed.append(stemmed_words)
```

```
snow_stemmed
```

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

Snowball Stemmer List Comprehension

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


Exp-3.ipynb
File Edit View Insert Runtime Tools Help Last saved at 7:53PM
Comment Share
+ Code + Text
Connect

Lancaster Stemmer

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

```
lancaster_stemmed
```

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

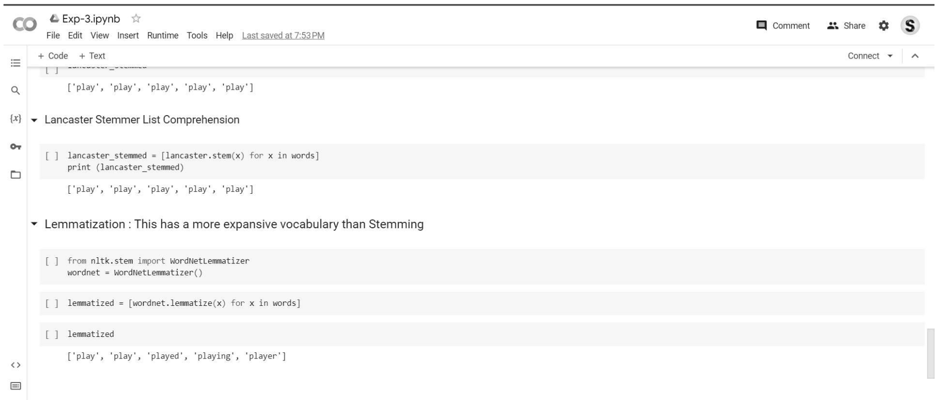
Lancaster Stemmer List Comprehension

```
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

```
from nltk.stem import WordNetLemmatizer
wordnet = WordNetLemmatizer()
```


Exp-3.ipynb
File Edit View Insert Runtime Tools Help Last saved at 7:53PM
Comment Share
+ Code + Text
Connect

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

Lancaster Stemmer List Comprehension

```
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

```
from nltk.stem import WordNetLemmatizer
wordnet = WordNetLemmatizer()
```

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

```
lemmatized
```

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

Conclusion: