Name: Sanket S. Fulzele

Roll No: 371018

PRN: 22110728

Div: A

Objective:

Perform various pre-processing tasks like tokenization, stemming,lemmatization, stop word removal.

NLP preprocessing Steps

In natural language processing (NLP), preprocessing refers to the tasks and techniques used to prepare text data for analysis and modelling. Here's an introduction to various preprocessing tasks commonly used in NLP:

1. Tokenization: Breaking text into smaller units, typically words or tokens. This step is crucial for further analysis as it allows the algorithm to work with individual elements of the text.
2. Lowercasing: Converting all text to lowercase. This helps in standardizing the text and treating words with different cases as the same, reducing the vocabulary size.
3. Stop word Removal: Removing common words like "and," "the," "is," etc., which do not carry significant meaning and can be noise in the analysis.
4. Punctuation Removal: Eliminating punctuation marks from the text, as they often do not contribute to the semantics of the text and can be distracting for models.
5. Normalization: Converting words to their base or root form. This includes tasks like stemming (reducing words to their base form by removing suffixes) and lemmatization (reducing words to their base form based on the dictionary).
6. Spell Checking and Correction: Correcting spelling errors in the text using algorithms that compare words against a known dictionary or language model.
7. Token Filtering: Removing tokens based on specific criteria, such as removing numbers, special characters, or specific types of words based on domain knowledge.
8. Part-of-Speech (POS) Tagging: Assigning grammatical tags to each word in the text, such as noun, verb, adjective, etc. This information is useful for tasks like named entity recognition and syntactic analysis.
9. Entity Recognition: Identifying and categorizing entities like names of people, organizations, locations, etc., in the text.
10. Vectorization: Converting text data into numerical vectors, often using techniques like Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), or word embeddings like Word2Vec or GloVe. This step is essential for machine learning models to process text data. These preprocessing tasks are often applied in combination, depending on the specific NLP task and the characteristics of the text data being analysed

```
!pip install indic-nlp-library
```

```
!pip install spacy
```

```
!pip install  pyPDF2
```

```
import nltk
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer, PorterStemmer
from nltk.probability import FreqDist
from nltk.corpus import stopwords
from PyPDF2 import PdfReader as pdf
```

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

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
def extract_text_from_pdf(pdf_path):
    text = ""
    with open(pdf_path, 'rb') as file:
        reader = pdf(file)
        num_pages = len(reader.pages)
        for page_num in range(num_pages):
            page = reader.pages[page_num]
            text += page.extract_text()
    return text
```

```
pdf_path = "/content/drive/MyDrive/PHY 1.pdf"
corpus=extract_text_from_pdf(pdf_path)
```

```
# text = "Natural language processing (NLP) is a field of computer science, artificial intelligence (AI), and li

tokens = word_tokenize(corpus)
```

```
lemmatizer = WordNetLemmatizer()
lemmatized_tokens = [lemmatizer.lemmatize(token) for token in tokens]
```

```
stemmer = PorterStemmer()
stemmed_tokens = [stemmer.stem(token) for token in tokens]
```

```
# Remove stopwords
stop_words = set(stopwords.words('english'))
filtered_tokens = [token for token in tokens if token.lower() not in stop_words]
```

```
# Count total unique words
total_unique_words = len(set(tokens))
```

```
# Calculate Type-Token Ratio (TTR)
type_token_ratio = len(set(tokens)) / len(tokens)
```

```
# print("Original Text:")
# print(text)
print("\nTokenization:")
print(tokens)
print("\nLemmatization:")
print(lemmatized_tokens)
print("\nStemming:")
print(stemmed_tokens)
print("\nTotal Unique Words:", total_unique_words)
print("\nType-Token Ratio (TTR):", type_token_ratio)
```

```
Tokenization:
['PHYSICS', 'PBL', 'NAME', ':', 'SANKET', 'SANJAY', 'FULZELE', 'Div', '.', 'H', 'ROLL', 'No', ':', '842', 'B

Lemmatization:
['PHYSICS', 'PBL', 'NAME', ':', 'SANKET', 'SANJAY', 'FULZELE', 'Div', '.', 'H', 'ROLL', 'No', ':', '842', 'B

Stemming:
['physic', 'pbl', 'name', ':', 'sanket', 'sanjay', 'fulzel', 'div', '.', 'h', 'roll', 'no', ':', '842', 'bat

Total Unique Words: 487

Type-Token Ratio (TTR): 0.45471521942110177
```

```
!python -m spacy download es_core_news_sm
```

```
import spacy
from collections import Counter

nlp = spacy.load("es_core_news_sm")

text = "La rápida zorra marrón salta sobre el perro perezoso."

doc = nlp(text)
tokens_lemmas = [(token.text, token.lemma_) for token in doc]


from nltk.stem import SnowballStemmer
stemmer = SnowballStemmer(language='spanish')
stemmed_tokens = [stemmer.stem(token.text) for token in doc]

unique_words = set(token.text for token in doc)

type_token_ratio = len(unique_words) / len(tokens_lemmas)

print("Tokenization and Lemmatization:")
print(tokens_lemmas)
print("\nStemming:")
print(stemmed_tokens)
print("\nTotal Unique Words:", len(unique_words))
print("Type Token Ratio:", type_token_ratio)
```

```
Tokenization and Lemmatization:
[('La', 'el'), ('rápida', 'rápido'), ('zorra', 'zorra'), ('marrón', 'marrón'), ('salta', 'salta'), ('sobre',

Stemming:
['la', 'rap', 'zorr', 'marron', 'salt', 'sobr', 'el', 'perr', 'perez', '.']

Total Unique Words: 10
Type Token Ratio: 1.0
```