NLP

Text Preprocessing

To prepare the text data for the model building we perform text preprocessing.

Remove HTML Tags and URLs
HTML tags and URLs are not important in model building. We have to remove them.
For removing HTML tags and URLs we use regex. RegEx is a package which checks if a string contains the specified search pattern.

Punctuation Removal:
In this step, all the punctuations from the text are removed.If we did not remove punctuation then punctuation is also considered one word. ‘string’ library of Python contains some pre-defined list of punctuations such as ‘!”#$%&'()*+,-./:;?@[\]^_`{|}~’

Lowering the text:
Here the text is converted into lower case.
string.lower()

Tokenization:
In this step, the text is split into smaller units. We can use either sentence tokenization or word tokenization based on our problem statement. Sentence tokenisation breaks the data text into sentences and word tokenisation breaks the sentence into individual word tokens. Tokenization can be done using NLTK Library.

```
import nltk
sentence = "The First sentence is about Python. The Second: about Django. You can learn Python,Django and Data Ananlysis here. "
tokens = nltk.sent_tokenize(sentence)
print (tokens)
```

```
output: ['The First sentence is about Python.', 'The Second: about Django.', 'You can learn Python,Django and Data Ananlysis here.']
```

Non-English tokenization:

```
import nltk
sentence = "The First sentence is about Python. The Second: about Django. You can learn Python,Django and Data Ananlysis here. "
tokens = nltk.sent_tokenize(sentence)
print (tokens)
```

```
output: ['Wie geht es Ihnen?', 'Gut, danke.']
```

Stop word removal:
Stopwords are the commonly used words which carry less or no meaning and are removed from the text as they do not add any value to the analysis.
NLTK library consists of a list of words that are considered stopwords for the English language. Some of them are : [i, me, my, myself, we, our, ours, ourselves, you, you're, you've, you'll, you'd, your, yours, yourself,etc]

Stemming:
It is also known as the text standardization step where the words are stemmed or diminished to their root/base form. **For example**, words like 'programmer', 'programming, 'program' will be stemmed to 'program'.
But the **disadvantage** of stemming is that it stems the words such that its root form loses the meaning or it is not diminished to a proper English word.
crazy-> crazi
available-> avail

```
from nltk.stem import PorterStemmer
from nltk.tokenize import sent_tokenize, word_tokenize

ps = PorterStemmer()
print(ps.stem("goose"))
print(ps.stem("geese"))
```

Lemmatization:
It stems the word but makes sure that it does not lose its meaning. Lemmatization has a pre-defined dictionary (wordnet) that stores the context of words and checks the word in the dictionary while diminishing.

```
from nltk.stem import WordNetLemmatizer

lem = WordNetLemmatizer()

print(lem.lemmatize("goose"))
print(lem.lemmatize("geese"))
print(lem.lemmatize("better", pos="a"))
```

The difference between Stemming and Lemmatization can be understood with the example provided

Original Word	After Stemming	After Lemmatization
goose	goos	goose
geese	gees	goose