Practical No. : 07

1. Extract Sample document and apply following document preprocessing methods: Tokenization, POS Tagging, stop words removal, Stemming and Lemmatization.
2. Create representation of document by calculating Term Frequency and Inverse Document Frequency.

Sample Sentences

In [1]:
```
Sample_Sentences = "I played the play playfully as the players were playing in the play with playfullness"
```

Tokenization

In [2]:
```
import nltk
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\shreyash\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

Out[2]:
```
True
```

In [ ]:
```

```

In [11]:
```
from nltk.tokenize import sent_tokenize
```

In [12]:
```
sentences = sent_tokenize(Sample_Sentences)
```

In [13]:
```
sentences
```

Out[13]:
```
['I played the play playfully as the players were playing in the play with playfullness']
```

In [3]:
```
from nltk import word_tokenize, sent_tokenize
sentences = sent_tokenize(Sample_Sentences)
tokenized_words = [word_tokenize(sentence) for sentence in sentences]
print('sentences words: ',sentences )
print('Tokenized words:', tokenized_words)
```

```
sentences words: ['I played the play playfully as the players were playing in the play with playfullness']
Tokenized words: [['I', 'played', 'the', 'play', 'playfully', 'as', 'the', 'players', 'were', 'playing', 'in', '
the', 'play', 'with', 'playfullness']]
```

POS Tagging

In [4]:
```
from nltk import pos_tag
tokenized_words = word_tokenize(Sample_Sentences)
pos_tags = pos_tag(tokenized_words)
print("Tagging Parts of Speech:", pos_tags)
```

```
Tagging Parts of Speech: [('I', 'PRP'), ('played', 'VBD'), ('the', 'DT'), ('play', 'NN'), ('playfully', 'RB'), (
'as', 'IN'), ('the', 'DT'), ('players', 'NNS'), ('were', 'VBD'), ('playing', 'VBG'), ('in', 'IN'), ('the', 'DT')
, ('play', 'NN'), ('with', 'IN'), ('playfullness', 'NN')]
```

Stop-Words Removal

In [5]:
```
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))
filtered_tokens = [word for word in tokenized_words if word.lower() not in stop_words]
print("Filtered Tokens after Stop Words Removal:", filtered_tokens)
```

```
Filtered Tokens after Stop Words Removal: ['played', 'play', 'playfully', 'players', 'playing', 'play', 'playful
lness']
```

Stemming

In [6]:
```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()
stemmed_tokens = [stemmer.stem(word) for word in filtered_tokens]
print("Stemmed Tokens:", stemmed_tokens)
```

```
Stemmed Tokens: ['play', 'play', 'play', 'player', 'play', 'play', 'playful']
```

Lemmatization

In [7]:
```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
lemmatized_tokens = [lemmatizer.lemmatize(word) for word in filtered_tokens]
print("Lemmatized Tokens:", lemmatized_tokens)
```

```
Lemmatized Tokens: ['played', 'play', 'playfully', 'player', 'playing', 'play', 'playfullness']
```

2) Create representation of document by calculating Term Frequency and Inverse Document Frequency.

In [8]:
```
preprocessed_text = ' '.join(lemmatized_tokens)
```

In [9]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer()
tfidf_representation = tfidf_vectorizer.fit_transform([preprocessed_text])

print("Preprocessed Text:", preprocessed_text)
print("\nTF-IDF Representation:")
print(tfidf_representation.toarray())
```

```
Preprocessed Text: played play playfully player playing play playfullness

TF-IDF Representation:
[[0.66666667 0.33333333 0.33333333 0.33333333 0.33333333 0.33333333]]
```