ASSIGNMENT 7

Text Analytics

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.

In [1]:
```
import nltk
nltk.download("punkt")
nltk.download("stopwords")
nltk.download("wordnet")
nltk.download("averaged_perceptron_tagger")
```

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

Out[1]:
```
True
```

In [5]:
```
from nltk import word_tokenize, sent_tokenize
```

In [17]:
```
corpus = "An octopus has three hearts, and two of them stop beating when it swims."
```

In [19]:
```
print(word_tokenize(corpus))
print(sent_tokenize(corpus))
```

```
['An', 'octopus', 'has', 'three', 'hearts', ',', 'and', 'two', 'of', 'them', 'stop',
'beating', 'when', 'it', 'swims', '.']
['An octopus has three hearts, and two of them stop beating when it swims.']
```

In [11]:
```
from nltk import pos_tag
```

In [13]:
```
tokens = word_tokenize(corpus)
print(pos_tag(tokens))
```

```
[('An', 'DT'), ('octopus', 'NN'), ('has', 'VBZ'), ('three', 'CD'), ('hearts', 'NN
S'), (',', ','), ('and', 'CC'), ('two', 'CD'), ('of', 'IN'), ('them', 'PRP'), ('sto
p', 'VB'), ('beating', 'NN'), ('when', 'WRB'), ('it', 'PRP'), ('swims', 'VBZ'),
('.', '.')]
```

In [21]:
```
from nltk.corpus import stopwords
stop_words = set(stopwords.words("english"))
```

In [23]:
```
tokens = word_tokenize(corpus)
cleaned_tokens = []
for token in tokens:
  if (token not in stop_words):
    cleaned_tokens.append(token)
print(cleaned_tokens)
```

```
['An', 'octopus', 'three', 'hearts', ',', 'two', 'stop', 'beating', 'swims', '.']
```

In [29]:
```
from nltk.stem import PorterStemmer
```

In [31]:
```
stemmer = PorterStemmer()
```

In [33]:
```
stemmed_tokens = []
for token in cleaned_tokens:
  stemmed = stemmer.stem(token)
  stemmed_tokens.append(stemmed)
print(stemmed_tokens)
```

```
['an', 'octopu', 'three', 'heart', ',', 'two', 'stop', 'beat', 'swim', '.']
```

In [35]:
```
from nltk.stem import WordNetLemmatizer
```

In [37]:
```
lemmatizer = WordNetLemmatizer()
```

In [39]:
```
lemmatized_tokens = []
for token in cleaned_tokens:
  lemmatized = lemmatizer.lemmatize(token)
  lemmatized_tokens.append(lemmatized)
print(lemmatized_tokens)
```

```
['An', 'octopus', 'three', 'heart', ',', 'two', 'stop', 'beating', 'swim', '.']
```

In [41]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [43]:
```
corpus = [
    "They were discussing various strategies to improve their performance in the co
    ]
```

In [45]:
```
vectorizer = TfidfVectorizer()
```

In [47]:
```
matrix = vectorizer.fit(corpus)
matrix.vocabulary_
```

Out[47]:
```
{'they': 22,
 'were': 28,
 'discussing': 8,
 'various': 25,
 'strategies': 18,
 'to': 23,
 'improve': 12,
 'their': 21,
 'performance': 15,
 'in': 13,
 'the': 20,
 'competition': 6,
 'sun': 19,
 'was': 27,
 'setting': 16,
 'casting': 3,
 'beautiful': 2,
 'colors': 5,
 'across': 0,
 'sky': 17,
 'children': 4,
 'eagerly': 9,
 'waiting': 26,
 'for': 10,
 'arrival': 1,
 'of': 14,
 'ice': 11,
 'cream': 7,
 'truck': 24}
```

In [49]:
```
tfidf_matrix = vectorizer.transform(corpus)
print(tfidf_matrix)
```

```
  (0, 28)	0.23006945204561577
  (0, 25)	0.30251368128649075
  (0, 23)	0.30251368128649075
  (0, 22)	0.30251368128649075
  (0, 21)	0.30251368128649075
  (0, 20)	0.1786694534059618
  (0, 18)	0.30251368128649075
  (0, 15)	0.30251368128649075
  (0, 13)	0.30251368128649075
  (0, 12)	0.30251368128649075
  (0, 8)	0.30251368128649075
  (0, 6)	0.30251368128649075
  (1, 27)	0.326245442256267
  (1, 20)	0.3853716274664007
  (1, 19)	0.326245442256267
  (1, 17)	0.326245442256267
  (1, 16)	0.326245442256267
  (1, 5)	0.326245442256267
  (1, 3)	0.326245442256267
  (1, 2)	0.326245442256267
  (1, 0)	0.326245442256267
  (2, 28)	0.21325889644076784
  (2, 26)	0.2804098208422736
  (2, 24)	0.2804098208422736
  (2, 20)	0.4968436720596348
  (2, 14)	0.2804098208422736
  (2, 11)	0.2804098208422736
  (2, 10)	0.2804098208422736
  (2, 9)	0.2804098208422736
  (2, 7)	0.2804098208422736
  (2, 4)	0.2804098208422736
  (2, 1)	0.2804098208422736
```

In [57]:
```
print(vectorizer.get_feature_names_out())
```

```
['across' 'arrival' 'beautiful' 'casting' 'children' 'colors'
 'competition' 'cream' 'discussing' 'eagerly' 'for' 'ice' 'improve' 'in'
 'of' 'performance' 'setting' 'sky' 'strategies' 'sun' 'the' 'their'
 'they' 'to' 'truck' 'various' 'waiting' 'was' 'were']
```