In [4]:
```
import nltk
nltk.download('punkt')
```

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

Out[4]:
```
True
```

In [6]:
```
from nltk import word_tokenize,sent_tokenize
sent = "The cookies were baked fresh this morning. They smell nice."
print("word Tokenize: ",word_tokenize(sent))
print('\n')
print("sentence Tokenize: :", sent_tokenize(sent))
```

```
word Tokenize:  ['The', 'cookies', 'were', 'baked', 'fresh', 'this', 'morning', '.',
'They', 'smell', 'nice', '.']


sentence Tokenize: : ['The cookies were baked fresh this morning.', 'They smell nic
e.']
```

In [21]:
```
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Yash Bahekar\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger.zip.
```

Out[21]:
```
True
```

In [22]:
```
from nltk import pos_tag
token = word_tokenize(sent)
tagged = pos_tag(token)
print("POS Tagged: ", tagged)
```

```
POS Tagged:  [('The', 'DT'), ('cookies', 'NNS'), ('were', 'VBD'), ('baked', 'VBN'),
('fresh', 'JJ'), ('this', 'DT'), ('morning', 'NN'), ('.', '.'), ('They', 'PRP'), ('s
mell', 'VBP'), ('nice', 'RB'), ('.', '.')]
```

In [16]:
```
nltk.download("stopwords")
```

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

Out[16]:
```
True
```

In [25]:
```
print("STOP WORDS REMOVAL\n")
from nltk.corpus import stopwords
stop_words = stopwords.words("english")
token = word_tokenize(sent)
cleaned_token = []
for word in token:
    if word not in stop_words:
        cleaned_token.append(word)
```

```
print("Before: ", token)
print("After: ", cleaned_token)
```

```
STOP WORDS REMOVAL

Before:  ['The', 'cookies', 'were', 'baked', 'fresh', 'this', 'morning', '.', 'The
y', 'smell', 'nice', '.']
After:  ['The', 'cookies', 'baked', 'fresh', 'morning', '.', 'They', 'smell', 'nic
e', '.']
```

In [28]:
```
print("STEMMING\n")

from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
words = token
stemmed = [stemmer.stem(word) for word in words]
print("Before Stemming: ", words)
print("After Stemming: ", stemmed)
```

```
STEMMING

Before Stemming:  ['The', 'cookies', 'were', 'baked', 'fresh', 'this', 'morning',
'.', 'They', 'smell', 'nice', '.']
After Stemming:  ['the', 'cooki', 'were', 'bake', 'fresh', 'thi', 'morn', '.', 'the
y', 'smell', 'nice', '.']
```

In [30]:
```
nltk.download('wordnet')
```

```
Lematization
[nltk_data] Downloading package wordnet to C:\Users\Yash
[nltk_data]     Bahekar\AppData\Roaming\nltk_data...
```

Out[30]:
```
True
```

In [34]:
```
print("Lematization\n")
from nltk.stem import WordNetLemmatizer

lemma = WordNetLemmatizer()
lemmas = []
for i in token:
    lem = lemma.lemmatize(i, pos='v')
    lemmas.append(lem)
print("Before Lemmatizing: ", token)
print("After Lemmatizing: ",lemmas)
```

```
Lematization

Before Lemmatizing:  ['The', 'cookies', 'were', 'baked', 'fresh', 'this', 'morning',
'.', 'They', 'smell', 'nice', '.']
After Lemmatizing:  ['The', 'cookies', 'be', 'bake', 'fresh', 'this', 'morning',
'.', 'They', 'smell', 'nice', '.']
```

In [36]:
```
print("TF-IDF\n")

from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer()
doc_1 = "The cookies were baked fresh this morning."
```

```
doc_2 = "They smell nice."
response = tfidf.fit_transform([doc_1,doc_2])

print("Vocabulary: ")
tfidf.vocabulary_
```

```
TF-IDF

Vocabulary:
```

Out[36]:
```
{'the': 6,
 'cookies': 1,
 'were': 9,
 'baked': 0,
 'fresh': 2,
 'this': 8,
 'morning': 3,
 'they': 7,
 'smell': 5,
 'nice': 4}
```

In [37]:
```
print(response)
```

```
 (0, 3)        0.3779644730092272
 (0, 8)        0.3779644730092272
 (0, 2)        0.3779644730092272
 (0, 0)        0.3779644730092272
 (0, 9)        0.3779644730092272
 (0, 1)        0.3779644730092272
 (0, 6)        0.3779644730092272
 (1, 4)        0.5773502691896257
 (1, 5)        0.5773502691896257
 (1, 7)        0.5773502691896257
```

In [ ]: