Text Analytics

In [ ]:

```
```

Tokenization

In [ ]:

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[59]:

```
True
```

In [ ]:

```
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("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.', 'T
hey smell nice.']
```

POS Tagging

In [ ]:

```
import nltk
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

Out[61]:

```
True
```

In [ ]:

```
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', 'N
N'), ('.', '.'), ('They', 'PRP'), ('smell', 'VBP'), ('nice', 'RB'),
('.', '.')]
```

Stop Words Removal

In [ ]:

```
  import nltk
  nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[63]:

```
True
```

In [ ]:

```
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)
```

```
Before:  ['The', 'cookies', 'were', 'baked', 'fresh', 'this', 'morni
ng', '.', 'They', 'smell', 'nice', '.']
After:  ['The', 'cookies', 'baked', 'fresh', 'morning', '.', 'They',
'smell', 'nice', '.']
```

Stemming

In [ ]:

```
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)
```

```
Before Stemming:  ['The', 'cookies', 'were', 'baked', 'fresh', 'thi
s', 'morning', '.', 'They', 'smell', 'nice', '.']
After Stemming:  ['the', 'cooki', 'were', 'bake', 'fresh', 'thi', 'm
orn', '.', 'they', 'smell', 'nice', '.']
```

Lematization

In [ ]:

```
import nltk
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[66]:

```
True
```

In [ ]:

```
from nltk.stem import LancasterStemmer, 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 )
```

```
Before Lemmatizing:  ['The', 'cookies', 'were', 'baked', 'fresh', 't
his', 'morning', '.', 'They', 'smell', 'nice', '.']
After Lemmatizing:  ['The', 'cookies', 'be', 'bake', 'fresh', 'thi
s', 'morning', '.', 'They', 'smell', 'nice', '.']
```

In [ ]:

```
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_
```

```
Vocabulary:
```

Out[68]:

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

In [ ]:

```
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 [ ]:

```
feature_names = tfidf.get_feature_names()
for col in response.nonzero()[1]:
  print(feature_names[col], ' - ', response[0, col])
```

```
morning  -  0.3779644730092272
this  -  0.3779644730092272
fresh  -  0.3779644730092272
baked  -  0.3779644730092272
were  -  0.3779644730092272
cookies  -  0.3779644730092272
the  -  0.3779644730092272
nice  -  0.0
smell  -  0.0
they  -  0.0

/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:
87: FutureWarning: Function get_feature_names is deprecated; get_fea
ture_names is deprecated in 1.0 and will be removed in 1.2. Please u
se get_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```