Name : Mohit Manish Bhavsar

Roll No : 20U437

Div : 4

```
import nltk

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from nltk import word_tokenize, sent_tokenize
from nltk.corpus import stopwords

nltk.download('stopwords')
```

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

```
True
```

```
text = "Natural Language Processing or NLP is a branch of artificial
intelligence that deals with the interaction between computers and humans
using the natural language. The ultimate objective of NLP is to read,
decipher, understand, and make sense of human languages in a manner that is
valuable. To this end, many different models, libraries, and methods have
been used to train machines to process text, understand it, make predictions
based on it, and even generate new text. The first step to training a model
is to obtain and preprocess the data. In this article, I will be going
through some of the most common steps to be followed with almost any dataset
before you can pass it as an input to a model."

words = word_tokenize(text, preserve_line=True)

print(words)
```

```
['Natural', 'Language', 'Processing', 'or', 'NLP', 'is', 'a', 'branch', 'of',
'artificial', 'intelligence', 'that', 'deals', 'with', 'the', 'interaction',
'between', 'computers', 'and', 'humans', 'using', 'the', 'natural',
'language.', 'The', 'ultimate', 'objective', 'of', 'NLP', 'is', 'to', 'read',
',', 'decipher', ',', 'understand', ',', 'and', 'make', 'sense', 'of',
'human', 'languages', 'in', 'a', 'manner', 'that', 'is', 'valuable.', 'To',
'this', 'end', ',', 'many', 'different', 'models', ',', 'libraries', ',',
'and', 'methods', 'have', 'been', 'used', 'to', 'train', 'machines', 'to',
'process', 'text', ',', 'understand', 'it', ',', 'make', 'predictions',
'based', 'on', 'it', ',', 'and', 'even', 'generate', 'new', 'text.', 'The',
'first', 'step', 'to', 'training', 'a', 'model', 'is', 'to', 'obtain', 'and',
```

```
'preprocess', 'the', 'data.', 'In', 'this', 'article', ',', 'I', 'will',
'be', 'going', 'through', 'some', 'of', 'the', 'most', 'common', 'steps',
'to', 'be', 'followed', 'with', 'almost', 'any', 'dataset', 'before', 'you',
'can', 'pass', 'it', 'as', 'an', 'input', 'to', 'a', 'model', '.']

print(len(words))

133

unique_words = list(set(words))

print(len(unique_words
         ))

91

sent = sent_tokenize(text)

len(sent)

5

sw = stopwords.words('english')

filtered_text = [i for i in unique_words if i not in sw]
print(filtered_text)
print(len(filtered_text))

['data.', 'NLP', 'almost', 'Natural', 'used', 'languages', 'read', 'first',
'preprocess', 'text', 'followed', 'sense', 'based', 'To', 'objective',
'model', 'In', 'artificial', 'step', 'steps', 'understand', 'intelligence',
'many', 'libraries', 'text.', 'dataset', 'even', 'generate', 'new',
'Processing', 'The', 'going', 'common', 'make', 'models', 'train',
'valuable.', 'deals', 'human', 'article', 'decipher', 'predictions',
'Language', 'natural', 'different', 'I', 'language.', 'using', '.',
'methods', 'machines', 'branch', ',', 'end', 'training', 'ultimate',
'obtain', 'humans', 'pass', 'process', 'computers', 'input', 'manner',
'interaction']
64

import string

pun = string.punctuation
print(pun)

!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~

print(filtered_text)
print(len(filtered_text))

['data.', 'NLP', 'almost', 'Natural', 'used', 'languages', 'read', 'first',
'preprocess', 'text', 'followed', 'sense', 'based', 'To', 'objective',
'model', 'In', 'artificial', 'step', 'steps', 'understand', 'intelligence',
```

```
'many', 'libraries', 'text.', 'dataset', 'even', 'generate', 'new',
'Processing', 'The', 'going', 'common', 'make', 'models', 'train',
'valuable.', 'deals', 'human', 'article', 'decipher', 'predictions',
'Language', 'natural', 'different', 'I', 'language.', 'using', '.',
'methods', 'machines', 'branch', ',', 'end', 'training', 'ultimate',
'obtain', 'humans', 'pass', 'process', 'computers', 'input', 'manner',
'interaction']
64
```

```
punctuation_removal = [i for i in filtered_text if i not in pun]

print(punctuation_removal)
print(len(punctuation_removal))
```

```
['data.', 'NLP', 'almost', 'Natural', 'used', 'languages', 'read', 'first',
'preprocess', 'text', 'followed', 'sense', 'based', 'To', 'objective',
'model', 'In', 'artificial', 'step', 'steps', 'understand', 'intelligence',
'many', 'libraries', 'text.', 'dataset', 'even', 'generate', 'new',
'Processing', 'The', 'going', 'common', 'make', 'models', 'train',
'valuable.', 'deals', 'human', 'article', 'decipher', 'predictions',
'Language', 'natural', 'different', 'I', 'language.', 'using', 'methods',
'machines', 'branch', 'end', 'training', 'ultimate', 'obtain', 'humans',
'pass', 'process', 'computers', 'input', 'manner', 'interaction']
62
```

```
from nltk.stem import PorterStemmer

text = punctuation_removal

ps = PorterStemmer()

stem_text = [ps.stem(i) for i in text]

stem_text
```

```
['data.',
 'nlp',
 'almost',
 'natur',
 'use',
 'languag',
 'read',
 'first',
 'preprocess',
 'text',
 'follow',
 'sens',
 'base',
 'to',
 'object',
 'model',
 'in',
```

```
 'artifici',
 'step',
 'step',
 'understand',
 'intellig',
 'mani',
 'librari',
 'text.',
 'dataset',
 'even',
 'gener',
 'new',
 'process',
 'the',
 'go',
 'common',
 'make',
 'model',
 'train',
 'valuable.',
 'deal',
 'human',
 'articl',
 'deciph',
 'predict',
 'languag',
 'natur',
 'differ',
 'i',
 'language.',
 'use',
 'method',
 'machin',
 'branch',
 'end',
 'train',
 'ultim',
 'obtain',
 'human',
 'pass',
 'process',
 'comput',
 'input',
 'manner',
 'interact']
```

```
from nltk.stem import WordNetLemmatizer
```

```
wl =  WordNetLemmatizer()
```

```
lem_text = [wl.lemmatize(i) for i in text]
```

```
lem_text
```

```
['data.',
 'NLP',
 'almost',
 'Natural',
 'used',
 'language',
 'read',
 'first',
 'preprocess',
 'text',
 'followed',
 'sense',
 'based',
 'To',
 'objective',
 'model',
 'In',
 'artificial',
 'step',
 'step',
 'understand',
 'intelligence',
 'many',
 'library',
 'text.',
 'dataset',
 'even',
 'generate',
 'new',
 'Processing',
 'The',
 'going',
 'common',
 'make',
 'model',
 'train',
 'valuable.',
 'deal',
 'human',
 'article',
 'decipher',
 'prediction',
 'Language',
 'natural',
 'different',
 'I',
```

```
 'language.',
 'using',
 'method',
 'machine',
 'branch',
 'end',
 'training',
 'ultimate',
 'obtain',
 'human',
 'pas',
 'process',
 'computer',
 'input',
 'manner',
 'interaction']
```

```
d1 = "Natural Language Processing or NLP is a branch of artificial
intelligence that deals with the interaction between computers and humans
using the natural language"
d2 = "The ultimate objective of NLP is to read, decipher, understand, and
make sense of human languages in a manner that is valuable"
d3 = "To this end, many different models, libraries, and methods have been
used to train machines to process text, understand it, make predictions based
on it, and even generate new text"

doc = [d1,d2,d3]

from sklearn.feature_extraction.text import TfidfVectorizer

tf = TfidfVectorizer(smooth_idf=True)

r3 = tf.fit_transform(doc).toarray()

print(tf.get_feature_names())
print(len(tf.get_feature_names()))
```

```
['and', 'artificial', 'based', 'been', 'between', 'branch', 'computers',
'deals', 'decipher', 'different', 'end', 'even', 'generate', 'have', 'human',
'humans', 'in', 'intelligence', 'interaction', 'is', 'it', 'language',
'languages', 'libraries', 'machines', 'make', 'manner', 'many', 'methods',
'models', 'natural', 'new', 'nlp', 'objective', 'of', 'on', 'or',
'predictions', 'process', 'processing', 'read', 'sense', 'text', 'that',
'the', 'this', 'to', 'train', 'ultimate', 'understand', 'used', 'using',
'valuable', 'with']
54
```

```
print(r3)
```

```
[[0.11817991 0.20009598 0.         0.         0.20009598 0.20009598
  0.20009598 0.20009598 0.         0.         0.         0.
  0.         0.         0.         0.20009598 0.         0.20009598
```

```
  0.20009598 0.15217815 0.         0.40019195 0.         0.
  0.         0.         0.         0.         0.         0.
  0.40019195 0.         0.15217815 0.         0.15217815 0.
  0.20009598 0.         0.         0.20009598 0.         0.
  0.         0.15217815 0.3043563  0.         0.         0.
  0.         0.         0.         0.20009598 0.         0.20009598]
 [0.13751474 0.         0.         0.         0.         0.
  0.         0.         0.23283269 0.         0.         0.
  0.         0.         0.23283269 0.         0.23283269 0.
  0.         0.35415052 0.         0.         0.23283269 0.
  0.         0.17707526 0.23283269 0.         0.         0.
  0.         0.         0.17707526 0.23283269 0.35415052 0.
  0.         0.         0.         0.         0.23283269 0.23283269
  0.         0.17707526 0.17707526 0.         0.17707526 0.
  0.23283269 0.17707526 0.         0.         0.23283269 0.        ]
 [0.20035941 0.         0.16961899 0.16961899 0.         0.
  0.         0.         0.         0.16961899 0.16961899 0.16961899
  0.16961899 0.16961899 0.         0.         0.         0.
  0.         0.         0.33923797 0.         0.         0.16961899
  0.16961899 0.12899961 0.         0.16961899 0.16961899 0.16961899
  0.         0.16961899 0.         0.         0.         0.16961899
  0.         0.16961899 0.16961899 0.         0.         0.
  0.33923797 0.         0.         0.16961899 0.38699883 0.16961899
  0.         0.12899961 0.16961899 0.         0.         0.        ]]
```

```
res = pd.DataFrame(r3,columns=tf.get_feature_names())
res.head()
```

```
        and  artificial     based      been   between    branch  computers  \
0  0.118180    0.200096  0.000000  0.000000  0.200096  0.200096   0.200096
1  0.137515    0.000000  0.000000  0.000000  0.000000  0.000000   0.000000
2  0.200359    0.000000  0.169619  0.169619  0.000000  0.000000   0.000000

      deals  decipher  different  ...       the      this        to     train
\
0  0.200096  0.000000   0.000000  ...  0.304356  0.000000  0.000000  0.000000
1  0.000000  0.232833   0.000000  ...  0.177075  0.000000  0.177075  0.000000
2  0.000000  0.000000   0.169619  ...  0.000000  0.169619  0.386999  0.169619

   ultimate  understand      used     using  valuable      with
0  0.000000    0.000000  0.000000  0.200096  0.000000  0.200096
1  0.232833    0.177075  0.000000  0.000000  0.232833  0.000000
2  0.000000    0.129000  0.169619  0.000000  0.000000  0.000000

[3 rows x 54 columns]
```

```
res.shape
```

```
(3, 54)
```