practical7

April 2, 2025

[1]:
```
import nltk
```

[2]:
```
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
```

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

[2]:
```
True
```

[3]:
```
#Initialize the text
text = "Hey Suhani, I just finished training a machine learning model on our␣
 ↪dataset. The accuracy improved to 92% after hyperparameter tuning! Let's␣
 ↪discuss the next steps in our meeting tomorrow."
```

[4]:
```
from nltk.tokenize import sent_tokenize
tokenized_text= sent_tokenize(text)
print(tokenized_text)
```

```
['Hey Suhani, I just finished training a machine learning model on our
dataset.', 'The accuracy improved to 92% after hyperparameter tuning!', 'Let's
discuss the next steps in our meeting tomorrow.']
```

[5]:
```
from nltk.tokenize import word_tokenize
tokenized_word=word_tokenize(text)
```

```
print(tokenized_word)
```

```
['Hey', 'Suhani', ',', 'I', 'just', 'finished', 'training', 'a', 'machine',
'learning', 'model', 'on', 'our', 'dataset', '.', 'The', 'accuracy', 'improved',
'to', '92', '%', 'after', 'hyperparameter', 'tuning', '!', 'Let', '’', 's',
'discuss', 'the', 'next', 'steps', 'in', 'our', 'meeting', 'tomorrow', '.']
```

[7]:
```
import re
from nltk.corpus import stopwords
stop_words=set(stopwords.words("english"))
print(stop_words)
text= "Data science is an interdisciplinary field that uses scientific methods,␣
 ↪processes, and algorithms to extract knowledge from data."
text= re.sub('[^a-zA-Z]', ' ',text)
tokens = word_tokenize(text.lower())
filtered_text=[]
for w in tokens:
    if w not in stop_words:
        filtered_text.append(w)
print("Tokenized Sentence:",tokens)
print("Filterd Sentence:",filtered_text)
```

```
{'their', 'very', 'with', 'of', 'now', 'were', 'couldn', "didn't", "i'm", 'll',
'which', 'before', 'own', 'from', 'under', 'about', 'above', 'yours', 'against',
'them', 'our', 'him', 're', 't', 'more', 'or', 'haven', 'in', 'y', 've', 'into',
'but', 'being', 'until', 'themselves', 'how', "you'll", 'both', "you've", 'd',
"hadn't", 'hadn', 'do', 'he', "we're", 'whom', 'hasn', 'nor', 'weren', 'wouldn',
'just', 'on', 'again', "they're", 'the', 'is', 'should', 'are', 'during', 'for',
'each', 'hers', 'been', "he'll", 'while', "we'll", "i'd", 'does', 'down',
'herself', 'all', 'because', "she's", 'myself', 'they', "weren't", "you're",
'and', "needn't", 'between', 'don', 'as', 'her', 'who', 's', 'aren', 'those',
'will', 'this', 'm', 'ma', 'itself', "don't", 'i', 'out', 'by', "he's", 'doesn',
"he'd", 'yourself', "should've", 'there', 'off', 'its', 'once', "shan't", 'so',
"isn't", 'some', 'what', 'that', 'such', 'when', 'if', 'his', "haven't", "i'll",
"it's", 'wasn', 'needn', 'was', 'didn', 'at', 'yourselves', 'ourselves',
"you'd", 'not', "it'll", 'most', 'up', 'these', 'won', 'your', "mightn't",
'mustn', 'himself', 'few', 'having', "she'd", 'any', 'no', 'than', 'below',
'ours', "wouldn't", 'only', 'through', "shouldn't", "that'll", 'where', "we've",
"i've", 'theirs', "hasn't", "couldn't", "doesn't", 'ain', 'o', 'mightn', 'too',
'an', "she'll", "it'd", "they've", 'shan', 'am', "aren't", "they'd", 'further',
'be', "mustn't", 'me', 'here', 'you', 'has', 'we', 'a', 'other', 'isn', 'had',
'to', 'same', 'she', "we'd", "they'll", 'after', 'it', "wasn't", 'did', 'doing',
'have', 'over', 'then', 'my', 'can', "won't", 'why', 'shouldn'}
Tokenized Sentence: ['data', 'science', 'is', 'an', 'interdisciplinary',
'field', 'that', 'uses', 'scientific', 'methods', 'processes', 'and',
'algorithms', 'to', 'extract', 'knowledge', 'from', 'data']
Filterd Sentence: ['data', 'science', 'interdisciplinary', 'field', 'uses',
```

```
'scientific', 'methods', 'processes', 'algorithms', 'extract', 'knowledge',
'data']
```

[8]:
```
from nltk.stem import PorterStemmer
ps = PorterStemmer()
e_words= ["play","playing","played","plays"]
for w in e_words:
    rootWord=ps.stem(w)
print(rootWord)
```

```
play
```

[9]:
```
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
wordnet_lemmatizer = WordNetLemmatizer()
text = "studies studying cries cry"
tokenization = word_tokenize(text)
for w in tokenization:
    print("Lemma for {} is {}".format(w, wordnet_lemmatizer.lemmatize(w)))
```

```
Lemma for studies is study
Lemma for studying is studying
Lemma for cries is cry
Lemma for cry is cry
```

[10]:
```
import nltk
from nltk.tokenize import word_tokenize
data = "The pink sweater fit her perfectly"
words = word_tokenize(data)
for word in words:
    print(nltk.pos_tag([word]))
```

```
[('The', 'DT')]
[('pink', 'NN')]
[('sweater', 'NN')]
[('fit', 'NN')]
[('her', 'PRP$')]
[('perfectly', 'RB')]
```

[ ]:
```

```