

zqbaubkn6

March 29, 2025

```
[ ]: #NAME: SHIVANI GADKARI  
#ROLL NO.: 13342
```

```
[ ]: #1. ALGORITHM FOR TOKENIZATION, POS TAGGING, STOP WORDS, STEMMING AND LEMMATIZATION:
```

```
[1]: pip install nltk
```

```
Requirement already satisfied: nltk in c:\users\sudam  
gadkari\anaconda3\lib\site-packages (3.9.1)  
Requirement already satisfied: click in c:\users\sudam  
gadkari\anaconda3\lib\site-packages (from nltk) (8.1.7)  
Requirement already satisfied: joblib in c:\users\sudam  
gadkari\anaconda3\lib\site-packages (from nltk) (1.4.2)  
Requirement already satisfied: regex>=2021.8.3 in c:\users\sudam  
gadkari\anaconda3\lib\site-packages (from nltk) (2024.9.11)  
Requirement already satisfied: tqdm in c:\users\sudam  
gadkari\anaconda3\lib\site-packages (from nltk) (4.66.5)  
Requirement already satisfied: colorama in c:\users\sudam  
gadkari\anaconda3\lib\site-packages (from click->nltk) (0.4.6)  
Note: you may need to restart the kernel to use updated packages.
```

```
[7]: #Download the required packages :  
import nltk  
nltk.download('punkt')  
nltk.download('stopwords')  
nltk.download('wordnet')  
nltk.download('averaged_perceptron_tagger')
```

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

[7]: True

```
[8]: #Initialize the text :
text= "Tokenization is the first step in text analytics. The process of
      ↪breaking down a text paragraph into smaller chunks such as words or
      ↪sentences is called Tokenization."
```

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

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

[10]: True

```
[22]: nltk.download('punkt_tab')
```

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

[22]: True

```
[19]: #Sentence Tokenization :
      from nltk.tokenize import sent_tokenize
      tokenized_text= sent_tokenize(text)
      print(tokenized_text)
```

```
['Tokenization is the first step in text analytics.', 'The process of breaking
down a text paragraph into smaller chunks such as words or sentences is called
Tokenization.']
```

```
[23]: #Word Tokenization :
      from nltk.tokenize import word_tokenize
      tokenized_word=word_tokenize(text)
      print(tokenized_word)
```

```
['Tokenization', 'is', 'the', 'first', 'step', 'in', 'text', 'analytics', '.',
'The', 'process', 'of', 'breaking', 'down', 'a', 'text', 'paragraph', 'into',
'smaller', 'chunks', 'such', 'as', 'words', 'or', 'sentences', 'is', 'called',
'Tokenization', '.']
```

```
[24]: #Removing Punctuations and stop words :
# print stop words of English
from nltk.corpus import stopwords
stop_words=set(stopwords.words("english"))
print(stop_words)
```

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

```
[28]: import re
text= "How to remove stop words with NLTK library in Python?"
text= re.sub('[^a-zA-Z]', ' ',text) #re.sub used to remove non alphabetic
↳characters
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)
```

Tokenized Sentence: ['how', 'to', 'remove', 'stop', 'words', 'with', 'nltk', 'library', 'in', 'python']
 Filterd Sentence: ['remove', 'stop', 'words', 'nltk', 'library', 'python']

```
[31]: #Perform Stemming(reducing words to its root form) :
from nltk.stem import PorterStemmer
```

```
e_words= ["wait", "waiting", "waited", "waits"]
ps =PorterStemmer()
for w in e_words:
    rootWord=ps.stem(w)
print(rootWord)
```

wait

```
[40]: #Performing lemmatization(covert the root to its baase):
from nltk.stem import WordNetLemmatizer
import nltk

# Initialize the lemmatizer
wordnet_lemmatizer = WordNetLemmatizer()

# Input text
text = "studies studying cries cry"

# Tokenize the text
tokenization = nltk.word_tokenize(text)

# Perform lemmatization on each token
for w in tokenization:
    print("Lemma for {}: {}".format(w, wordnet_lemmatizer.lemmatize(w)))
```

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

```
[47]: import nltk
nltk.download('punkt') # For tokenizing the text
nltk.download('averaged_perceptron_tagger') # For POS tagging
```

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

```
[47]: True
```

```
[51]: #Apply POS Tagging to text :
import nltk
from nltk.tokenize import word_tokenize
```

```

# Sample data
data = "The pink sweater fit her perfectly"

# Tokenize the text
words = word_tokenize(data)

# Perform POS tagging
for word in words:
    print(nltk.pos_tag([word]))

```

```

-----
LookupError                                Traceback (most recent call last)
Cell In[51], line 13
     11 # Perform POS tagging
     12 for word in words:
--> 13     print(nltk.pos_tag([word]))

File ~\anaconda3\Lib\site-packages\nltk\tag\__init__.py:168, in pos_tag(tokens,
    ↪tagset, lang)
     143 def pos_tag(tokens, tagset=None, lang="eng"):
     144     """
     145     Use NLTK's currently recommended part of speech tagger to
     146     tag the given list of tokens.
     (...)
     166     :rtype: list(tuple(str, str))
     167     """
--> 168     tagger = _get_tagger(lang)
     169     return _pos_tag(tokens, tagset, tagger, lang)

File ~\anaconda3\Lib\site-packages\nltk\tag\__init__.py:110, in _get_tagger(lang)
     108     tagger = PerceptronTagger(lang=lang)
     109 else:
--> 110     tagger = PerceptronTagger()
     111 return tagger

File ~\anaconda3\Lib\site-packages\nltk\tag\perceptron.py:183, in
    ↪PerceptronTagger.__init__(self, load, lang)
     181 self.classes = set()
     182 if load:
--> 183     self.load_from_json(lang)

File ~\anaconda3\Lib\site-packages\nltk\tag\perceptron.py:273, in
    ↪PerceptronTagger.load_from_json(self, lang)
     271 def load_from_json(self, lang="eng"):
     272     # Automatically find path to the tagger if location is not specified.
--> 273     loc = find(f"taggers/averaged_perceptron_tagger_{lang}/")

```

```

274     with open(loc + TAGGER_JSONS[lang]["weights"]) as fin:
275         self.model.weights = json.load(fin)

```

File ~\anaconda3\Lib\site-packages\nltk\data.py:579, in find(resource_name, paths)

```

577 sep = "*" * 70
578 resource_not_found = f"\n{sep}\n{msg}\n{sep}\n"
--> 579 raise LookupError(resource_not_found)

```

LookupError:

Resource `averaged_perceptron_tagger_eng` not found.
Please use the NLTK Downloader to obtain the resource:

```

>>> import nltk
>>> nltk.download('averaged_perceptron_tagger_eng')

```

For more information see: <https://www.nltk.org/data.html>

Attempted to load `taggers/averaged_perceptron_tagger_eng/`

Searched in:

- 'C:\\Users\\SUDAM GADKARI\\nltk_data'
- 'C:\\Users\\SUDAM GADKARI\\anaconda3\\nltk_data'
- 'C:\\Users\\SUDAM GADKARI\\anaconda3\\share\\nltk_data'
- 'C:\\Users\\SUDAM GADKARI\\anaconda3\\lib\\nltk_data'
- 'C:\\Users\\SUDAM GADKARI\\AppData\\Roaming\\nltk_data'
- 'C:\\nltk_data'
- 'D:\\nltk_data'
- 'E:\\nltk_data'

```

[52]: import nltk
nltk.download('punkt') # For word tokenization
nltk.download('averaged_perceptron_tagger') # For POS tagging

```

```

[nltk_data] Downloading package punkt to C:\Users\SUDAM
[nltk_data]       GADKARI\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]       C:\Users\SUDAM GADKARI\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!

```

[52]: True

```
[53]: import nltk
nltk.download('averaged_perceptron_tagger_eng')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data] C:\Users\SUDAM GADKARI\AppData\Roaming\nltk_data...
[nltk_data] Unzipping taggers\averaged_perceptron_tagger_eng.zip.
```

```
[53]: True
```

```
[54]: import nltk
from nltk.tokenize import word_tokenize

# Sample data
data = "The pink sweater fit her perfectly"

# Tokenize the text
words = word_tokenize(data)

# Perform POS tagging
for word in words:
    print(nltk.pos_tag([word]))
```

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

```
[ ]:
```