```
Start coding or generate with AI.
```

```
('1', 1) ('2', 4) ('1', 2) ('2', 1) ('3', 1) ('4', 1) ('2', 3)
```

NLTK vs SPACY ??

corpus = paragraph

TOKENIZATION

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
from nltk.tokenize import sent_tokenize #convert the paragraph into senteces
```

```
pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.9.1
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from
```

```
sent_tokenize(corpus)
```

```
import nltk
```

```
nltk.download('punkt_tab')
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
True
```

```
corpus = """ The gentle breeze swept through the meadow,
 carrying the soft scent of wildflowers and freshly cut grass.
 Birds chirped melodiously, their songs blending seamlessly with the
 rustling of leaves. A stream gurgled nearby, its clear water reflecting
  the azure sky above. Children laughed as they played, their carefree joy
   echoing across the field. An old oak tree stood tall, its branches providing
    a cool shade where an elderly couple sat, reminiscing about their youth.
    Time seemed to slow in this serene corner of nature, offering a brief
```

```
respite from the world’s chaos. Moments like these felt like treasures,
 simple yet profound."""


from nltk.tokenize import sent_tokenize


sent_tokenize(corpus)
```

```
[' The gentle breeze swept through the meadow,\n carrying the soft scent of
wildflowers and freshly cut grass.',
 'Birds chirped melodiously, their songs blending seamlessly with the \n rustling of
leaves.',
 'A stream gurgled nearby, its clear water reflecting\n  the azure sky above.',
 'Children laughed as they played, their carefree joy\n   echoing across the
field.',
 'An old oak tree stood tall, its branches providing\n    a cool shade where an
elderly couple sat, reminiscing about their youth.',
 'Time seemed to slow in this serene corner of nature, offering a brief\n
respite from the world’s chaos.',
 'Moments like these felt like treasures,\n      simple yet profound.']
```

```
#paragraph into words or sentence into words


from nltk.tokenize import word_tokenize


word_tokenize(corpus)
```

```
['The',
 'gentle',
 'breeze',
 'swept',
 'through',
 'the',
 'meadow',
 ',',
 'carrying',
 'the',
 'soft',
 'scent',
 'of',
 'wildflowers',
 'and',
 'freshly',
 'cut',
 'grass',
 '.',
 'Birds',
 'chirped',
 'melodiously',
 ',',
 'their',
 'songs',
 'blending',
 'seamlessly',
 'with',
 'the',
```

```
 'rustling',
 'of',
 'leaves',
 '.',
 'A',
 'stream',
 'gurgled',
 'nearby',
 ',',
 'its',
 'clear',
 'water',
 'reflecting',
 'the',
 'azure',
 'sky',
 'above',
 '.',
 'Children',
 'laughed',
 'as',
 'they',
 'played',
 ',',
 'their',
 'carefree',
 'joy',
 'echoing',
 'across',.
```

```
from nltk.tokenize import wordpunct_tokenize  #puncutations
```

```
wordpunct_tokenize(corpus)
```

```
['The',
 'gentle',
 'breeze',
 'swept',
 'through',
 'the',
 'meadow',
 ',',
 'carrying',
 'the',
 'soft',
 'scent',
 'of',
 'wildflowers',
 'and',
 'freshly',
 'cut',
 'grass',
 '.',
 'Birds',
 'chirped',
 'melodiously',
 ',',
 'their',
 'songs',
```

```
 'blending',
 'seamlessly',
 'with',
 'the',
 'rustling',
 'of',
 'leaves',
 '.',
 'A',
 'stream',
 'gurgled',
 'nearby',
 ',',
 'its',
 'clear',
 'water',
 'reflecting',
 'the',
 'azure',
 'sky',
 'above',
 '.',
 'Children',
 'laughed',
 'as',
 'they',
 'played',
 ',',
 'their',
 'carefree',
 'joy',
 'echoing',
 'across',
```

```
from nltk.tokenize import TreebankWordTokenizer #. is not separate word but only last ful
```

```
TreebankWordTokenizer().tokenize(corpus)
```

```
['The',
 'gentle',
 'breeze',
 'swept',
 'through',
 'the',
 'meadow',
 ',',
 'carrying',
 'the',
 'soft',
 'scent',
 'of',
 'wildflowers',
 'and',
 'freshly',
 'cut',
 'grass.',
 'Birds',
 'chirped',
 'melodiously',
```

```
        ',',
        'their',
        'songs',
        'blending',
        'seamlessly',
        'with',
        'the',
        'rustling',
        'of',
        'leaves.',
        'A',
        'stream',
        'gurgled',
        'nearby',
        ',',
        'its',
        'clear',
        'water',
        'reflecting',
        'the',
        'azure',
        'sky',
        'above.',
        'Children',
        'laughed',
        'as',
        'they',
        'played',
        ',',
        'their',
        'carefree',
        'joy',
        'echoing',
        'across',
        'the',
        'field.',
        'An',
```

```
words = [
    "running", "flies", "better", "playing", "children", "teeth",
    "cacti", "bought", "mice", "talked", "geese", "swimming",
    "driving", "quickly", "happier", "thinking", "studying", "wolves"
]


#PORTSTEMMER
from nltk.stem import PorterStemmer


stem= PorterStemmer()


for i in words:
  print(i + " : " + stem.stem(i))
```

```
running : run
flies : fli
better : better
playing : play
```

```
children : children
teeth : teeth
cacti : cacti
bought : bought
mice : mice
talked : talk
geese : gees
swimming : swim
driving : drive
quickly : quickli
happier : happier
thinking : think
studying : studi
wolves : wolv
```

```
stem.stem("studying") #disadvantage of stemming
```

```
'studi'
```

```
from nltk.stem import RegexpStemmer
```

```
reg= RegexpStemmer('ing$|s$|e$|able$', min=4)
```

```
for i in words:
  print(i + "=>" +" "+ reg.stem(i))
```

```
running=> runn
flies=> flie
better=> better
playing=> play
children=> children
teeth=> teeth
cacti=> cacti
bought=> bought
mice=> mic
talked=> talked
geese=> gees
swimming=> swimm
driving=> driv
quickly=> quickly
happier=> happier
thinking=> think
studying=> study
wolves=> wolve
```

```
reg.stem("studying") #we solve the above problem
```

```
'study'
```

```
from nltk.stem import SnowballStemmer
```

```
j=SnowballStemmer('english')
```

```
for i in words:
  print(i+"=>"+ j.stem(i))
```

```
running=>run
flies=>fli
better=>better
playing=>play
children=>children
teeth=>teeth
cacti=>cacti
bought=>bought
mice=>mice
talked=>talk
geese=>gees
swimming=>swim
driving=>drive
quickly=>quick
happier=>happier
thinking=>think
studying=>studi
wolves=>wolv
```

```
j.stem('studying'),j.stem('thinking')
```

```
('studi', 'think')
```

```
#lemmaitzation solve above problems we get root words time taking (Q&A,text summary,chatb
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

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

```
lemm=WordNetLemmatizer()
```

```
for i in words:
  print(i+"=>"+lemm.lemmatize(i,pos='v'))
```

```
running=>run
flies=>fly
better=>better
playing=>play
children=>children
teeth=>teeth
cacti=>cacti
bought=>buy
mice=>mice
talked=>talk
geese=>geese
swimming=>swim
driving=>drive
quickly=>quickly
happier=>happier
```

```
thinking=>think
studying=>study
wolves=>wolves
```

```
lemm.lemmatize('studying',pos='v'),lemm.lemmatize('goes',pos='n')
```

```
('study', 'go')
```

```
Pos - Noun n
Verb v
Adjective a
Adverb r
Adverb r
```

```
#STOPWORDS
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')
new="""
    "Shared values of democracy, liberty, and equality form the foundation of the India-U
    "Economic ties highlighted with American businesses thriving in India and Indian comp
    "Trade and capital flow between India and the U.S. generates jobs in both nations.",
    "A unified global response to terrorism is essential; terrorism must be delegitimized
    "India is committed to sustainable development and combating climate change by promot
    "India’s ancient belief emphasizes living in harmony with nature, aligning with globa
    "Defense partnership has grown significantly with joint military exercises and increa
    "Indian-American community praised for contributions in various fields, including tec
    "Cultural exchanges and the diaspora foster deeper understanding and collaboration be
    "India and the U.S. share a vision for global stability and prosperity through strate
    "Strengthened ties between the nations contribute to peace, economic growth, and secu
    "Optimism expressed for the future of the India-U.S. relationship as a force for glob
    "Both nations committed to upholding democratic values and addressing global challeng
    "The enduring friendship between India and the U.S. is built on shared ideals and mut
    "Prime Minister Modi's speech was met with standing ovations, reflecting strong bipar
```

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

```
stopwords.words('english')
```

```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
```

```
        "you'd",
        'your',
        'yours',
        'yourself',
        'yourselves',
        'he',
        'him',
        'his',
        'himself',
        'she',
        "she's",
        'her',
        'hers',
        'herself',
        'it',
        "it's",
        'its',
        'itself',
        'they',
        'them',
        'their',
        'theirs',
        'themselves',
        'what',
        'which',
        'who',
        'whom',
        'this',
        'that',
        "that'll",
        'these',
        'those',
        'am',
        'is',
        'are',
        'was',
        'were',
        'be',
        'been',
        'being',
        'have',
        'has',
        'had',
        'having',
        'do',
        'does',
```

```
  from nltk.stem import PorterStemmer
```

```
s=PorterStemmer()
```

```
up=nltk.sent_tokenize(new)
```

```
for i in range(len(up)):
  words=nltk.word_tokenize(up[i])
```

```
  words=[s.stem(word) for word in words if word not in set(stopwords.words('english'))]
  up[i]=' '.join(words)
```

Double-click (or enter) to edit

```
print(up)
```

```
['``  share valu democraci , liberti , equal form foundat india-u. .', 'relationship .
```

```
from nltk.stem import WordNetLemmatizer
lem=WordNetLemmatizer()
```

```
for i in range(len(up)):
  words=nltk.word_tokenize(up[i])
  words=[lem.lemmatize(word,pos='v') for word in words if word not in set(stopwords.words
  up[i]=' '.join(words)
```

```
print(up)
```

```
['``  share valu democraci , liberti , equal form foundat india-u . .', 'relationship
```

PART OF SPEECH TAGGING

```
ok="""
    "Shared values of democracy, liberty, and equality form the foundation of the India-U
    "Economic ties highlighted with American businesses thriving in India and Indian comp
    "Trade and capital flow between India and the U.S. generates jobs in both nations.",
    "A unified global response to terrorism is essential; terrorism must be delegitimized
    "India is committed to sustainable development and combating climate change by promot
    "India’s ancient belief emphasizes living in harmony with nature, aligning with globa
    "Defense partnership has grown significantly with joint military exercises and increa
    "Indian-American community praised for contributions in various fields, including tec
    "Cultural exchanges and the diaspora foster deeper understanding and collaboration be
    "India and the U.S. share a vision for global stability and prosperity through strate
    "Strengthened ties between the nations contribute to peace, economic growth, and secu
    "Optimism expressed for the future of the India-U.S. relationship as a force for glob
    "Both nations committed to upholding democratic values and addressing global challeng
    "The enduring friendship between India and the U.S. is built on shared ideals and mut
    "Prime Minister Modi's speech was met with standing ovations, reflecting strong bipar
```

```
import nltk
it=nltk.sent_tokenize(ok)
```

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

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

```
print(it)
```

```
['\n    "Shared values of democracy, liberty, and equality form the foundation of the
```

```
for i in range(len(it)):
  words=nltk.word_tokenize(it[i])
  words=nltk.pos_tag(words)
  print(words)
```

```
[('``', '``'), ('Shared', 'VBN'), ('values', 'NNS'), ('of', 'IN'), ('democracy', 'NN'
[('relationship', 'NN'), ('.', '.')]
[('``', '``'), (',', ','), ('``', '``'), ('Economic', 'NNP'), ('ties', 'NNS'), ('high
[('economy', 'NN'), ('.', '.')]
[('``', '``'), (',', ','), ('``', '``'), ('Trade', 'NNP'), ('and', 'CC'), ('capital',
[('``', '``'), (',', ','), ('``', '``'), ('A', 'DT'), ('unified', 'JJ'), ('global', '
[('``', '``'), (',', ','), ('``', '``'), ('India', 'NNP'), ('is', 'VBZ'), ('committed
[('``', '``'), (',', ','), ('``', '``'), ('India', 'NNP'), ('’', 'NNP'), ('s', 'VBD')
[('``', '``'), (',', ','), ('``', '``'), ('Defense', 'NNP'), ('partnership', 'NN'), (
[('``', '``'), (',', ','), ('``', '``'), ('Indian-American', 'JJ'), ('community', 'NN
[('``', '``'), (',', ','), ('``', '``'), ('Cultural', 'JJ'), ('exchanges', 'NNS'), ('
[('``', '``'), (',', ','), ('``', '``'), ('India', 'NNP'), ('and', 'CC'), ('the', 'DT
[('``', '``'), (',', ','), ('``', '``'), ('Strengthened', 'VBD'), ('ties', 'NNS'), ('
[('``', '``'), (',', ','), ('``', '``'), ('Optimism', 'NN'), ('expressed', 'VBN'), ('
[('``', '``'), (',', ','), ('``', '``'), ('Both', 'DT'), ('nations', 'NNS'), ('commit
[('``', '``'), (',', ','), ('``', '``'), ('The', 'DT'), ('enduring', 'VBG'), ('friend
[('``', '``'), (',', ','), ('``', '``'), ('Prime', 'NNP'), ('Minister', 'NNP'), ('Mod
```

EXAMPle pos_tag()

```
print(nltk.pos_tag("i am going to watch tajmahal".split()))
```

name entity recognition

```
par="Civilization is approximately 6,000 years old,tracing back to the emergence of organ
```

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

```
nltk.download('averaged_perceptron_tagger_eng')
nltk.download('maxent_ne_chunker_tab')
nltk.download('words')
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
[nltk_data] Downloading package maxent_ne_chunker_tab to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping chunkers/maxent_ne_chunker_tab.zip.
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Unzipping corpora/words.zip.
True
```

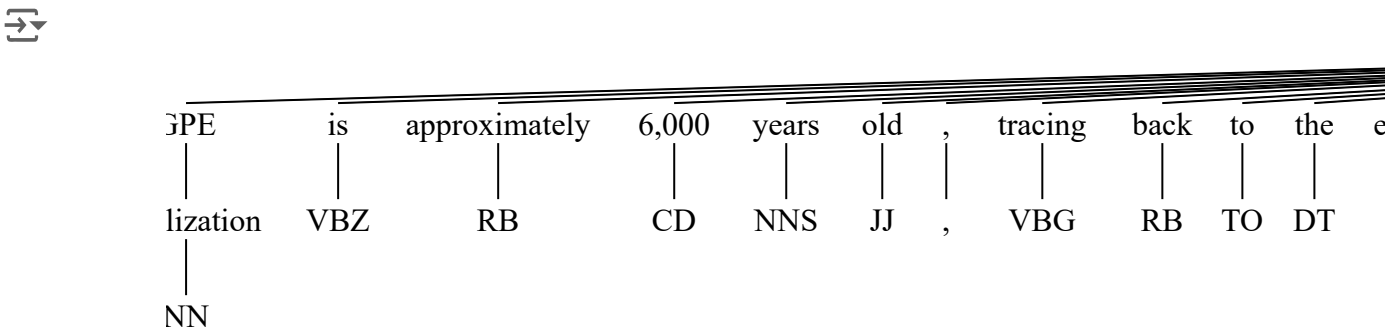
```
import nltk
nltk.pos_tag(nltk.word_tokenize(par))
```

```
[('Civilization', 'NN'),
 ('is', 'VBZ'),
 ('approximately', 'RB'),
 ('6,000', 'CD'),
 ('years', 'NNS'),
 ('old', 'JJ'),
 (',', ','),
 ('tracing', 'VBG'),
 ('back', 'RB'),
 ('to', 'TO'),
 ('the', 'DT'),
 ('emergence', 'NN'),
 ('of', 'IN'),
 ('organized', 'JJ'),
 ('societies', 'NNS'),
 ('in', 'IN'),
 ('Mesopotamia.These', 'JJ'),
 ('early', 'JJ'),
 ('civilizations', 'NNS'),
 ('laid', 'VBD'),
 ('the', 'DT'),
 ('foundation', 'NN'),
 ('for', 'IN'),
 ('human', 'JJ'),
 ('cultural', 'JJ'),
 (',', ','),
 ('social', 'JJ'),
 (',', ','),
 ('and', 'CC'),
 ('technological', 'JJ'),
 ('development', 'NN'),
 ('.', '.')]
```

Double-click (or enter) to edit

```
import svgling
```

```
nltk.ne_chunk(nltk.pos_tag(nltk.word_tokenize(par)))
```



ENCODING(ONE HOT)

advantage : easy to implement

disadvantage : sparse matrix causes overfitiing

anavailability if fixed size I/P

NO Semantic meaning is getting captured

Out of vocabulary

```
 #BAG OF WORDS(binary or normal)
 #1:lower case all words and also remove stopwords
 #2:calculate the vocabulary frequency(maintain a order either increasing or decreasing)
 #3: make those vocabs a feature
```

```
#ADVantage : simple,intutive and also we get fixed size of inputs
#disadvanatge : sparse matrix (overfitting) ,ordering of words getting changed,out of voc
```

```
#TF-IDF(term freq-inverse document freq)
#TF=(no of repetiton  words in sentence)/(no of words in sentence)
#IDF=log(no of senteces/ no of senteces contain the word)
#ADVantage :intutive,fixed size i/p,word importance getting captured
#DIsadvatage:
```

```
#Word embeddings:based on frequency (BOW,OHE,TFIDF)and based on deep learning(continuos b
```

```
#WORD2VEC(DEEP LEARNING)
#1:FEATURE representation
#2:COSINE similarity
```

```
pip install gensim
```

```
Requirement already satisfied: gensim in /usr/local/lib/python3.10/dist-packages (4.3
Requirement already satisfied: numpy<2.0,>=1.18.5 in /usr/local/lib/python3.10/dist-p
Requirement already satisfied: scipy<1.14.0,>=1.7.0 in /usr/local/lib/python3.10/dist
Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.10/dist-pa
Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from
```

```
import gensim
```

```
from gensim.models import Word2Vec, KeyedVectors
```

```
import gensim.downloader as api
wv = api.load('word2vec-google-news-300')
vec = wv['king']
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.