```
import pandas as  pd

a=  pd.read_csv("/content/Corona_NLP_test.csv")

a
```

```
      UserName  ScreenName             Location     TweetAt  \
0            1       44953                  NYC  02-03-2020
1            2       44954          Seattle, WA  02-03-2020
2            3       44955                  NaN  02-03-2020
3            4       44956          Chicagoland  02-03-2020
4            5       44957  Melbourne, Victoria  03-03-2020
...        ...         ...                  ...         ...
3793      3794       48746            Israel ??  16-03-2020
3794      3795       48747       Farmington, NM  16-03-2020
3795      3796       48748        Haverford, PA  16-03-2020
3796      3797       48749                  NaN  16-03-2020
3797      3798       48750  Arlington, Virginia  16-03-2020

                                          OriginalTweet
Sentiment
0     TRENDING: New Yorkers encounter empty supermar...  Extremely
Negative
1     When I couldn't find hand sanitizer at Fred Me...
Positive
2     Find out how you can protect yourself and love...  Extremely
Positive
3     #Panic buying hits #NewYork City as anxious sh...
Negative
4     #toiletpaper #dunnypaper #coronavirus #coronav...
Neutral
...                                                 ...
...
3793  Meanwhile In A Supermarket in Israel -- People...
Positive
3794  Did you panic buy a lot of non-perishable item...
Negative
3795  Asst Prof of Economics @cconces was on @NBCPhi...
Neutral
3796  Gov need to do somethings instead of biar je r...  Extremely
Negative
3797  I and @ForestandPaper members are committed to...  Extremely
Positive

[3798 rows x 6 columns]
```

```
a.columns
```

```
Index(['UserName', 'ScreenName', 'Location', 'TweetAt',
'OriginalTweet',
```

```
       'Sentiment'],
      dtype='object')
```

```
import nltk
from nltk import sent_tokenize
from nltk import word_tokenize
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
True
```

```
text = str(a['OriginalTweet'].head(5))
```

```
type(text)
```

```
str
```

```
text
```

```
{"type":"string"}
```

```
# Word Tokenization
tokens_words = nltk.word_tokenize(text)
print(tokens_words)
```

```
['0', 'TRENDING', ':', 'New', 'Yorkers', 'encounter', 'empty',
'supermar', '...', '1', 'When', 'I', 'could', "n't", 'find', 'hand',
'sanitizer', 'at', 'Fred', 'Me', '...', '2', 'Find', 'out', 'how',
'you', 'can', 'protect', 'yourself', 'and', 'love', '...', '3', '#',
'Panic', 'buying', 'hits', '#', 'NewYork', 'City', 'as', 'anxious',
'sh', '...', '4', '#', 'toiletpaper', '#', 'dunnypaper', '#',
'coronavirus', '#', 'coronav', '...', 'Name', ':', 'OriginalTweet',
',', 'dtype', ':', 'object']
```

```
# Part of Speech
token = nltk.word_tokenize(text)
print("Part of Speech: ", nltk.pos_tag(token))
```

```
Part of Speech:  [('0', 'CD'), ('TRENDING', 'NN'), (':', ':'), ('New',
'NNP'), ('Yorkers', 'NNPS'), ('encounter', 'VBP'), ('empty', 'JJ'),
('supermar', 'NN'), ('...', ':'), ('1', 'CD'), ('When', 'WRB'), ('I',
'PRP'), ('could', 'MD'), ("n't", 'RB'), ('find', 'VB'), ('hand',
```

```
'NN'), ('sanitizer', 'NN'), ('at', 'IN'), ('Fred', 'NNP'), ('Me',
'NNP'), ('...', ':'), ('2', 'CD'), ('Find', 'NNP'), ('out', 'RP'),
('how', 'WRB'), ('you', 'PRP'), ('can', 'MD'), ('protect', 'VB'),
('yourself', 'PRP'), ('and', 'CC'), ('love', 'VB'), ('...', ':'),
('3', 'CD'), ('#', '#'), ('Panic', 'NNP'), ('buying', 'VBG'), ('hits',
'NNS'), ('#', '#'), ('NewYork', 'NNP'), ('City', 'NNP'), ('as', 'IN'),
('anxious', 'JJ'), ('sh', 'NN'), ('...', ':'), ('4', 'CD'), ('#',
'#'), ('toiletpaper', 'JJ'), ('#', '#'), ('dunnypaper', 'JJ'), ('#',
'#'), ('coronavirus', 'NN'), ('#', '#'), ('coronav', 'NN'), ('...',
':'), ('Name', 'NN'), (':', ':'), ('OriginalTweet', 'NN'), (',', ','),
('dtype', 'NN'), (':', ':'), ('object', 'NN')]
```

```
# Lemmatizer
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
text2 = "Find out how you can protect yourself and loved ones from
#coronavirus. ?"
word_list = nltk.word_tokenize(text2)
print(word_list)
```

```
['Find', 'out', 'how', 'you', 'can', 'protect', 'yourself', 'and',
'loved', 'ones', 'from', '#', 'coronavirus', '.', '?']
```

```
lemmatized_output = ' '.join([lemmatizer.lemmatize(w) for w in
word_list])
print(lemmatized_output)
```

```
Find out how you can protect yourself and loved one from # coronavirus
. ?
```

```
# Stemmer
from nltk.stem import PorterStemmer
ps = PorterStemmer()
word = ("getting")
ps.stem(word)
```

```
{"type":"string"}
```

```
# Stop WordsWordsWordsWordsWordsWordsWordsWords
from nltk.corpus import stopwords
print(stopwords.words('english'))
# random sentecnce with lot of stop words
sample_text = "Oh man, this is pretty cool. We will do more such
things."
text_tokens = word_tokenize(sample_text)

tokens_without_sw = [word for word in text_tokens if not word in
stopwords.words('english')]

print(text_tokens)
print(tokens_without_sw)
```

```
['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', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or',
'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with',
'about', 'against', 'between', 'into', 'through', 'during', 'before',
'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out',
'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once',
'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both',
'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor',
'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't',
'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now',
'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't",
'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn',
"hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma',
'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan',
"shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren',
"weren't", 'won', "won't", 'wouldn', "wouldn't"]
['Oh', 'man', ',', 'this', 'is', 'pretty', 'cool', '.', 'We', 'will',
'do', 'more', 'such', 'things', '.']
['Oh', 'man', ',', 'pretty', 'cool', '.', 'We', 'things', '.']
```