In [1]:
```
import nltk
from nltk.corpus import brown
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from nltk import pos_tag
```

In [30]:
```
text=brown.raw(brown.fileids()[9])
sentences=sent_tokenize(text)
newsentence=text.replace('/n', ' ')
newsentence=newsentence.replace('/', ' ')
words = word_tokenize(newsentence)
```

In [3]:
```
stop_words=set(stopwords.words('english'))
filtered_words = [word for word in words if word.lower() not in stop_words]
```

In [31]:
```
stemmer = PorterStemmer()
stemmed_words = [stemmer.stem(word) for word in filtered_words]
```

In [32]:
```
lemmatizer = WordNetLemmatizer()
lemmatized_words = [lemmatizer.lemmatize(word) for word in filtered_words]
```

In [33]:
```
pos_tags = pos_tag(words)
```

In [18]:
```
import spacy
from spacy import displacy
```

In [24]:
```
NER = spacy.load('en_core_web_sm')
```

In [29]:
```
text0=NER(newsentence)
for word in text0.ents:
    print(word.text,word.label_)
```

```
ben PERSON
pps ORG
Monday DATE
Charles p E. p Raymond p ORG
District n-tl ORG
Ierulli p ORG
Desmond p D. p Connall p ORG
ben PERSON
March DATE
31 CARDINAL
Ierulli p ORG
29 CARDINAL
ben PERSON
Portland GPE
November DATE
1959 DATE
pps ORG
Portland p-tl University n-tl ORG
Northwestern jj-tl College n-tl ORG
pps ORG
three CARDINAL
E. p M. p Martin p ORG
World n-tl Affairs ORG
Council n-tl ORG
Monday DATE
Martin p ORG
Washington GPE
D. p C. p ORG
13 cd months DATE
Multnomah p-tl ORG
pps ORG
dti ORG
350 CARDINAL
United vbn-tl FAC
States GPE
bez congenial ORG
pps ORG
pps ORG
pps ORG
jjt PERSON
Soviet n-tl NORP
Union n-tl ORG
pps ORG
East LOC
Germany GPE
15 cd years DATE
Soviet NORP
pps ORG
Martin p ORG
Americans NORP
a at decade DATE
society n we ppss ORG
pps ORG
United vbn-tl FAC
pps ORG
best jjt-hl PERSON
Martin p ORG
United vbn-tl FAC
bez more ql ORG
pps ORG
```

```
United vbn-tl FAC
pps ORG
Martin p ORG
one CARDINAL
Portland p school n board n ORG
Monday DATE
Portland GPE
board n ORG
pps ORG
Ralph p H. p Molvar p ORG
1409 DATE
SW n Maplecrest p Dr. n-tl ORG
ppss ORG
ppss ORG
n-tl PERSON
C. p Richard p Mears ORG
school n board n ORG
n-tl PERSON
Melvin p W. p Barnes p ORG
pps ORG
dti ORG
ppss PERSON
pps ORG
n . GPE
ppss ORG
pps ORG
dti ORG
dt PERSON
Molvar p ORG
pps ORG
n-tl PERSON
Barnes p ORG
Molvar p ORG
board n ORG
Jack p Lowe's ORG
pps ORG
pps ORG
pps ORG
board n ORG
bez going vbg ORG
pps ORG
Salem GPE
AP p-hl ORG
Tuesday DATE
Salem GPE
Mark p Hatfield p . ORG
Hatfield p ORG
Tuesday DATE
noon TIME
Monday DATE
Eugene PERSON
Emerald n-tl Empire n-tl ORG
Kiwanis p-tl Club n-tl ORG
pps ORG
Willamette p-tl University n-tl FAC
Republicans NORP
Thursday DATE
Salem GPE
p . GPE
Friday DATE
```

```
pps ORG
Portland GPE
Dean p Bryson p ORG
Multnomah p-tl County n-tl Circuit n-tl ORG
pps ORG
Republican p State n-tl ORG
Central jj-tl LOC
Committee n-tl ORG
Saturday DATE
Portland GPE
Washington-Oregon p ORG
Beaverton p-tl School n-tl District n-tl No ORG
48 CARDINAL
two CARDINAL
Monday DATE
two CARDINAL
January DATE
board n ORG
4 CARDINAL
3 CARDINAL
6-3-3 CARDINAL
8-4 CARDINAL
Board n ORG
Monday DATE
Nov. DATE
15 CARDINAL
581,000 MONEY
three CARDINAL
n-tl PERSON
Labor n-tl ORG
Arthur p Goldberg p ORG
Sunday DATE
Temple n-tl ORG
25 MONEY
n-tl PERSON
Wayne p L. p Morse p ORG
Aj n PERSON
7 cd p.m. TIME
n-tl PERSON
Goldberg p ORG
n-tl PERSON
Morse p ORG
Roosevelt p Hotel n-tl FAC
4:30 cd p.m. TIME
Sunday DATE
Blaine p Whipple p ORG
Democratic NORP
Oregon GPE
Tuesday DATE
Edith p Green p ORG
Al p Ullman p ORG
Labor n-tl ORG
Norman p Nilsen p ORG
Mayor n-tl PERSON
Terry p Schrunk p ORG
Democrats NORP
Oak n-tl-hl PERSON
Grove n-tl-hl PERSON
Three CARDINAL
Oak n-tl FAC
```

```
Lodge n-tl Water n-tl district n board n ORG
11 CARDINAL
Dec. p DATE
4 CARDINAL
8 cd a.m. TIME
8 cd p.m. TIME
Richard p Salter p ORG
Donald p Huffman p ORG
five-year DATE
William p Brod p PERSON
Barbara p Njust p ORG
Miles p C. p Bubenik p ORG
Frank p Lee p . PERSON
Five CARDINAL
Hugh p G. p Stout p . ORG
two-year DATE
James p Culbertson p ORG
Dwight p M. p Steeves p ORG
James p C. p Piersee p ORG
W.M. p Sexton p ORG
Theodore p ORG
p . GPE
Friday DATE
29th DATE
general jj council n ORG
Assemblies ns-tl FAC
Memorial jj-tl FAC
Coliseum p-tl ORG
The at council n ORG
16 CARDINAL
1966 DATE
Bible p . FAC
council n ORG
pps ORG
Bible p . FAC
The at-tl Assemblies ns-tl of in-tl God p WORK_OF_ART
days DATE
Bible p WORK_OF_ART
first ORDINAL
third ORDINAL
first ORDINAL
16 CARDINAL
third ORDINAL
six CARDINAL
Diety n-tl ORG
Jesus p Christ p ORG
Virgin n-tl FAC
Christ p

         ORG
cross n

         ORG
Super n-hl ORG
Friday DATE
afternoon TIME
the at Rev. p T. p F. p Zimmerman p ORG
second ORDINAL
two-year DATE
Assemblies GPE
```

```
pp$ PERSON
Springfield GPE
Mo. p . ORG
Friday DATE
Breakthrough n-tl WORK_OF_ART
two cd years DATE
Wednesday DATE
Rev. p ORG
Zimmerman p ORG
10-year DATE
Breakthrough n-tl WORK_OF_ART
first ORDINAL
two cd years DATE
first ORDINAL
two cd years' ns$ DATE
one CARDINAL
three CARDINAL
Americans NORP
church n ORG
100 cd million CARDINAL
pps ORG
Friday DATE
Church n-hl ORG
12,000 CARDINAL
daily DATE
United vbn-tl FAC
1-1 2 CARDINAL
pps ORG
35 cd years DATE
7,000 CARDINAL
Rev. p ORG
Brandt p ORG
one CARDINAL
10,000 CARDINAL
dt PRODUCT
1,000 CARDINAL
Illinois p ORG
200 CARDINAL
800 CARDINAL
Southern jj-tl LOC
England p ORG
60 CARDINAL
100 CARDINAL
Rhode p-tl Island n-tl GPE
pps ORG
Rev. p ORG
Brandt p ORG
8,000 CARDINAL
Assemblies PERSON
10 cd years DATE
dti ORG
pps ORG
this dt hour TIME
6-12 CARDINAL
U.S. GPE
n-tl PERSON
Charles p L. p Powell ORG
Monday DATE
Portland GPE
n-tl PERSON
```

```
Powell p ORG
md join GPE
Dwight p L. p Schwab p ORG
Philip p Weinstein p ORG
Weinstein PERSON
n-tl PERSON
Powell p ORG
Proof n-hl PERSON
Schwab PERSON
Weinstein PERSON
U.S. GPE
U.S. GPE
afternoon TIME
```

In [25]:
```
raw_text="The Indian Space Research Organisation or is the national space agency of Ir
text1=NER(raw_text)
```

In [27]:
```
for word in text1.ents:
    print(word.text,word.label_)
```

```
The Indian Space Research Organisation ORG
India GPE
Bengaluru GPE
Department of Space ORG
India GPE
ISRO ORG
DOS ORG
```

In [28]:
```
# -*- coding: utf-8 -*-
"""
Created July 2017

@author: arw
"""

# Training your own chunker using chunked treeband data - again made available in NLTK
# As before (with tagging) we first divide the data into training and testing sets
from nltk.corpus import treebank_chunk
data = treebank_chunk.chunked_sents()
train_data = data[0:3500]
test_data = data[3500:]
print(train_data[0])

simple_sentence = 'the brown fox jumped over the lazy dog'

# Can use tagger from package pattern.en if using Python 2.x
# from pattern.en import tag
# tagged_sentence = tag(sentence)

import nltk
from nltk.chunk import RegexpParser
tokens = nltk.word_tokenize(simple_sentence)
tagged_simple_sent = nltk.pos_tag(tokens)
print(tagged_simple_sent)

# We first define our grammars using regex pattern using the RegexpParser
# We can specify which patterns we want to segment in a sentence as *chunks*
chunk_grammar = """
NP: {<DT>?<JJ>*<NN.*>}
VP: {<VBD><IN>}
```

```
"""
rc = RegexpParser(chunk_grammar)
c = rc.parse(tagged_simple_sent)
print(c)

# We sometimes want to specify which patterns we DO NOT want to segment in a sentence
# so that we can *chunk* all the others
chink_grammar = """
NP: {<JJ|NN>+} # chunk only adjective-noun pair as NP
"""

rc = RegexpParser(chink_grammar)
c = rc.parse(tagged_simple_sent)
print(c)


# A more realistic grammar for chunking
grammar = """
NP: {<DT>?<JJ>?<NN.*>}
ADJP: {<JJ>}
ADVP: {<RB.*>}
PP: {<IN>}
VP: {<MD>?<VB.*>+}

"""

# And a more realistic sentence as input
sentence = 'the brown fox is quick and he may jump over the lazy dog'
tokens = nltk.word_tokenize(sentence)
tagged_sent = nltk.pos_tag(tokens)

rc = RegexpParser(grammar)
c = rc.parse(tagged_sent)
print(c)

print(rc.evaluate(test_data))
# The performance is not great!
# Why is this?


# We have acecss to a utility function tree2conlltags which extracts word, tag and
# chunk triples from annotated text
from nltk.chunk.util import tree2conlltags, conlltags2tree
# Let's take a slightly more typical sentence from our data
train_sent = train_data[7]
print(train_sent)

# We extract the POS and chunk tags using tree2conlltags function which returns a list
wtc = tree2conlltags(train_sent)
wtc

# We can 'reverse' this to output a shallow tree using the conlltags2tree function
tree = conlltags2tree(wtc)
print(tree)


# We can use these features to train a 'combined' chunker as we did for POS tagging
def conll_tag_chunks(chunk_sents):
  tagged_sents = [tree2conlltags(tree) for tree in chunk_sents]
  return [[(t, c) for (w, t, c) in sent] for sent in tagged_sents]
```

```
def combined_tagger(train_data, taggers, backoff=None):
    for tagger in taggers:
        backoff = tagger(train_data, backoff=backoff)
    return backoff

from nltk.tag import UnigramTagger, BigramTagger
from nltk.chunk import ChunkParserI

# We create a new class to use the word, POS and Chunk tag features to train a chunker
# that is able to 'backoff' from bigram to a unigram model as before
# Can you have another layer for trigram and back off to this model?
class NGramTagChunker(ChunkParserI):

  def __init__(self, train_sentences,
               tagger_classes=[UnigramTagger, BigramTagger]):
    train_sent_tags = conll_tag_chunks(train_sentences)
    self.chunk_tagger = combined_tagger(train_sent_tags, tagger_classes)

  def parse(self, tagged_sentence):
    if not tagged_sentence:
        return None
    pos_tags = [tag for word, tag in tagged_sentence]
    chunk_pos_tags = self.chunk_tagger.tag(pos_tags)
    chunk_tags = [chunk_tag for (pos_tag, chunk_tag) in chunk_pos_tags]
    wpc_tags = [(word, pos_tag, chunk_tag) for ((word, pos_tag), chunk_tag)
                     in zip(tagged_sentence, chunk_tags)]
    return conlltags2tree(wpc_tags)

# We call our new class and pass it the training data from the chunked treebank
ntc = NGramTagChunker(train_data)
print(ntc.evaluate(test_data))
# Now we get really good results on the data set
# Why?

# Let's try to visualize the chunk for the more realistic sample sentence
tree = ntc.parse(tagged_sent)
print(tree)
tree.draw()


# We now use our shallow parser on a larger 'Wall Street Journal' corpus
# SAQ 1. How big is it?
# SAQ 2. How much test data did we have before, and how much now?
from nltk.corpus import conll2000
wsj_data = conll2000.chunked_sents()
train_wsj_data = wsj_data[:7500]
test_wsj_data = wsj_data[7500:]
print(train_wsj_data[10])

# We first train our model on the training corpus
tc = NGramTagChunker(train_wsj_data)
# And then we test it on the test data
print(tc.evaluate(test_wsj_data))
```

```
(S
  (NP Pierre/NNP Vinken/NNP)
  ,/,
  (NP 61/CD years/NNS)
  old/JJ
  ,/,
  will/MD
  join/VB
  (NP the/DT board/NN)
  as/IN
  (NP a/DT nonexecutive/JJ director/NN Nov./NNP 29/CD)
  ./.)
[('the', 'DT'), ('brown', 'JJ'), ('fox', 'NN'), ('jumped', 'VBD'), ('over', 'IN'),
('the', 'DT'), ('lazy', 'JJ'), ('dog', 'NN')]
(S
  (NP the/DT brown/JJ fox/NN)
  (VP jumped/VBD over/IN)
  (NP the/DT lazy/JJ dog/NN))
(S
  the/DT
  (NP brown/JJ fox/NN)
  jumped/VBD
  over/IN
  the/DT
  (NP lazy/JJ dog/NN))
(S
  (NP the/DT brown/JJ fox/NN)
  (VP is/VBZ)
  (ADJP quick/JJ)
  and/CC
  he/PRP
  (VP may/MD jump/VB)
  (PP over/IN)
  (NP the/DT lazy/JJ dog/NN))
C:\Users\User\AppData\Local\Temp\ipykernel_16184\2397733014.py:68: DeprecationWarnin
g:
  Function evaluate() has been deprecated.  Use accuracy(gold)
  instead.
  print(rc.evaluate(test_data))
```

```
ChunkParse score:
    IOB Accuracy:  46.1%%
    Precision:     19.9%%
    Recall:        43.3%%
    F-Measure:     27.3%%
(S
  (NP A/DT Lorillard/NNP spokewoman/NN)
  said/VBD
  ,/,
  ``/``
  (NP This/DT)
  is/VBZ
  (NP an/DT old/JJ story/NN)
  ./.)
(S
  (NP A/DT Lorillard/NNP spokewoman/NN)
  said/VBD
  ,/,
  ``/``
  (NP This/DT)
  is/VBZ
  (NP an/DT old/JJ story/NN)
  ./.)
C:\Users\User\AppData\Local\Temp\ipykernel_16184\2397733014.py:124: DeprecationWarnin
g:
  Function evaluate() has been deprecated.  Use accuracy(gold)
  instead.
  print(ntc.evaluate(test_data))
ChunkParse score:
    IOB Accuracy:  97.2%%
    Precision:     91.4%%
    Recall:        94.3%%
    F-Measure:     92.8%%
(S
  (NP the/DT brown/JJ fox/NN)
  is/VBZ
  (NP quick/JJ)
  and/CC
  (NP he/PRP)
  may/MD
  jump/VB
  over/IN
  (NP the/DT lazy/JJ dog/NN))
(S
  (NP He/PRP)
  (VP reckons/VBZ)
  (NP the/DT current/JJ account/NN deficit/NN)
  (VP will/MD narrow/VB)
  (PP to/TO)
  (NP only/RB #/# 1.8/CD billion/CD)
  (PP in/IN)
  (NP September/NNP)
  ./.)
C:\Users\User\AppData\Local\Temp\ipykernel_16184\2397733014.py:146: DeprecationWarnin
g:
  Function evaluate() has been deprecated.  Use accuracy(gold)
  instead.
  print(tc.evaluate(test_wsj_data))
```

```
ChunkParse score:
    IOB Accuracy:  89.4%%
    Precision:     80.8%%
    Recall:        86.0%%
    F-Measure:     83.3%%
```

In [ ]: