

EXPERIMENT 7

Aim: Perform chunking for the given text input

Theory:

Chunking:

It involves dividing text into syntactically correlated words

For example:

'He ate an apple' can be divided as

[NP He] [VP ate] [NP an apple]

Each chunk has an open boundary and close boundary that delimit. The word groups as a non-recursive unit. This can be formally expressed by IOB prefixes

[B-NP -He] [B-VP ate]
NP VP

[B-NP an] [I-NP apple]
NP

chunk Types:

Its types are based on the syntactic category part

| Chunk Type | Tag Name |
|-----------------|----------|
| ① Noun | NP |
| ② Verb | VP |
| ③ Adverb | ADVP |
| ④ Adjectival | ADJP |
| ⑤ prepositional | PP |

Tokens POS chunk -Tags :

| | | |
|---------|-----|------|
| He | PRP | B-NP |
| are | VBD | B-NP |
| an | DT | B-NP |
| apple | NN | I-NP |
| to | B | B-VP |
| satisfy | VB | I-VP |
| his | B | B-NP |
| hunger | NN | I-NP |

Chunk Types

| Type | Definition |
|-------------|--|
| characters | individual characters within text |
| words | words separated by any amount of white spaces (spaces, tabs) within text |
| lines | Paragraphs separated by the any of several standard line endings (CR, LF, CRLF, etc) |
| Text items | portions of text separated by commas |
| List items | The individual item in a list |
| bytes | the bytes within binary data |
| occurrences | The text matches of defined pattern |
| matches | The text matches and text range of a defined pattern and its capture groups |

Conclusion:

Hence, we understood the process of chunking and its types and performed the chunking on the given input.

Code

```
# Importing the required libraries
import nltk
from nltk import pos_tag
from nltk import word_tokenize
from nltk import RegexpParser

text = input("Enter Sentence")

# Splitting the sentence into words
list_of_words = word_tokenize(text)

# Applying POS_tagging
tagged_words = pos_tag(list_of_words)
# Extracting the Noun Phrases
chunk_to_be_extracted = r"Chunk: {<DT>*<NNP>*<NN>*"

# Applying chunking to the text
chunkParser = nltk.chunk.RegexpParser(chunk_to_be_extracted)
chunked_sentence = chunkParser.parse(tagged_words)
# To view the NLTK tree
chunked_sentence.draw()
# To print the chunks extracted

print('Chunks obtained: \n')
for subtree in chunked_sentence.subtrees():
    if subtree.label() == 'Chunk':
        print(subtree)
```

Output

Giving in the text input

Enter Sentence

NLTK

File Zoom



```
In [14]: # Extracting the Noun Phrases
chunk_to_be_extracted = r''' Chunk: {<DT>*<NNP>*<NN>*} '''

# Applying chunking to the text
chunkParser = nltk.chunk.RegexpParser(chunk_to_be_extracted)
chunked_sentence = chunkParser.parse(tagged_words)
```

```
In [15]: # To view the NLTK tree
chunked_sentence.draw()
```

```
In [16]: # To print the chunks extracted

print('Chunks obtained: \n')
for subtree in chunked_sentence.subtrees():
    if subtree.label() == 'Chunk':
        print(subtree)
```

Chunks obtained:

```
(Chunk The/DT Air/NNP India/NNP flight/NN)
(Chunk Delhi/NNP)
(Chunk board/NN)
```

In []: