Vidyavardhini's College of Engineering and Technology, Vasai

Department of Computer Science & Engineering (Data Science)

AY: 2025-26

Class:	**BE-CSE(DS)**	**Semester:**	**VII**
Course Code:	**CSDOL7011**	**Course Name:**	**NLP Lab**

Name of Student:	**Sahil Salunke**
Roll No. :	**45**
Experiment No.:	**6**
Title of the Experiment:	**Performing Chunking and Named Entity Recognition using NLTK**
Date of Performance:	
Date of Submission:	

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations (BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty :

Signature :

Date :

Aim: To identify and extract syntactic phrases (chunks) and named entities from text using NLTK's chunking and NER functionalities.

Objective: To extract syntactic chunks and named entities using chunking and Named Entity Recognition techniques.

Tools Required:

1. Python (Jupyter Notebook or Google Colab)
2. nltk

Procedure:

1. Import required libraries:
    a. import nltk
    b. nltk.download('punkt')
    c. nltk.download('averaged_perceptron_tagger')
    d. nltk.download('maxent_ne_chunker')
    e. nltk.download('words')

2. Input or define a sentence:

Example: "Barack Obama was born in Hawaii and served as the 44th President of the United States."

3. Tokenize and POS-tag the sentence:
    a. tokens = nltk.word_tokenize(sentence)
    b. pos_tags = nltk.pos_tag(tokens)

4. Apply chunking:

Use regular expressions to define grammar rules.

a. chunk_grammar = "NP: {<DT>?<JJ>*<NN>}"
b. chunk_parser = nltk.RegexpParser(chunk_grammar)
c. chunked = chunk_parser.parse(pos_tags)
d. chunked.draw() # Optional: visualize the parse tree

5. Perform Named Entity Recognition:
    a. ner_tree = nltk.ne_chunk(pos_tags)
    b. ner_tree.draw() # Optional visualization

6. Extract named entities:

Traverse the NER tree and extract named entities like PERSON, ORGANIZATION, LOCATION.

Description of the Experiment:

In this experiment, students will implement chunking to group words into syntactic units (like noun phrases), and perform Named Entity Recognition (NER) to identify proper nouns such as names of people, places, or organizations. These are foundational steps in syntactic and information extraction tasks.

Detailed Description of the NLP Technique:

1. Chunking (Shallow Parsing):

Chunking segments and labels multi-token sequences, such as noun phrases (NP) or verb phrases (VP), without generating full parse trees.

Example:

a. Input: "The quick brown fox"
b. POS tags: [(The, DT), (quick, JJ), (brown, JJ), (fox, NN)]

c. Chunk: (NP The quick brown fox)

Uses regular expressions on POS tag sequences to define chunk patterns.

2. Named Entity Recognition (NER):

NER identifies and classifies named entities in text into predefined categories such as:

PERSON (e.g., "Barack Obama")

ORGANIZATION (e.g., "Google")

LOCATION (e.g., "India")

DATE, TIME, MONEY, etc.

NLTK's ne_chunk() uses a pre-trained Maximum Entropy classifier to identify named entities.

Importance of Chunking and NER:

a. Enhances understanding of text structure.
b. Crucial in tasks like question answering, information extraction, and document classification.

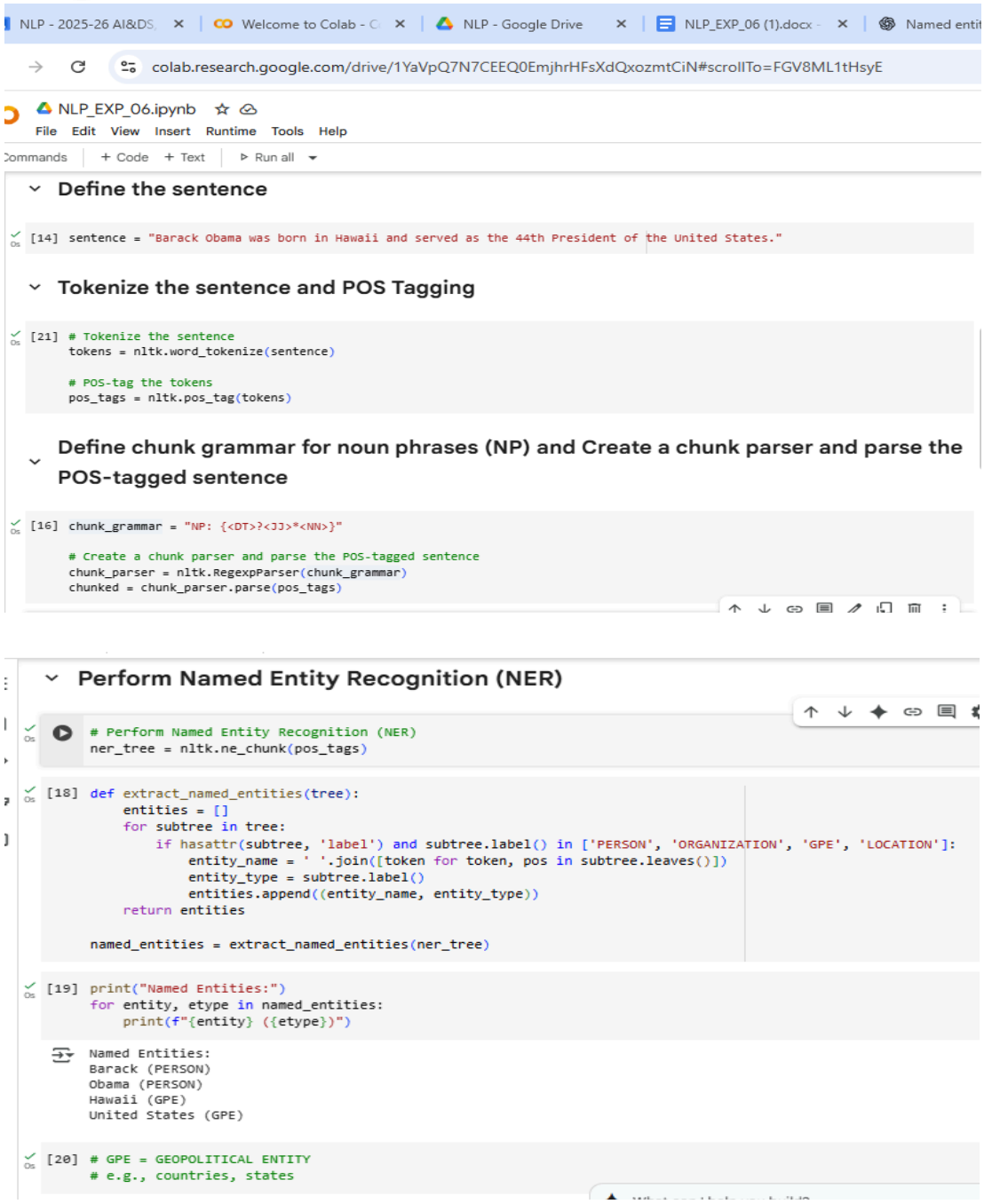
Code and Output:

Importing and Downloading libraries

```
[1]  import nltk
```

```
[13] nltk.download('punkt')
     nltk.download('punkt_tab')
     nltk.download('averaged_perceptron_tagger')
     nltk.download('averaged_perceptron_tagger_eng')
     nltk.download('maxent_ne_chunker')
     nltk.download('maxent_ne_chunker_tab')
     nltk.download('words')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[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!
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package maxent_ne_chunker is already up-to-date!
[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...
```



Conclusion:

When we run this code on the sentence, it successfully identifies important names like **Barack Obama** as a person and places like **Hawaii** and the United **States** as locations. The chunking step highlights simple noun phrases, which helps in understanding the sentence structure by

grouping words like "the 44th President." Overall, the named entity recognition does a great job picking out key real-world entities, giving us a clearer picture of who and what the sentence is talking about. This kind of analysis is really useful in tasks like information extraction or building smarter search engines.