Mrunal Labhe, A1 , 12

Practical 2:Aim:Write a Python NLTK program to split the text from genesis corpus and display it into a list of words. Remove the Punctuation and Stopwords from the given text and perform Stemming and POS tagging

Theory:

Natural Language Processing (NLP) enables computers to process, analyze, and understand human language. This lab employs the NLTK (Natural Language Toolkit) library in Python for core NLP tasks:

- Tokenization divides raw text into individual units (words).
- Stopword Removal eliminates common, non-informative words.
- Stemming reduces words to their root form for normalization.
- POS Tagging assigns parts of speech (like noun, verb) to each word.
- Dependency Parsing analyzes grammatical structure, identifying relationships within the sentence.

These steps are foundational for advanced text analysis, search, and AI language applications.

Code:

#importing all the dependencies

```
import nltk
from nltk.corpus import genesis, stopwords
from nltk.tokenize import word_tokenize , sent_tokenize
from nltk.stem import SnowballStemmer
from nltk import pos_tag
import string
nltk.download('genesis')
nltk.download('punkt_tab')

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

#loading the dataset :

```
genesis_corpus = genesis.raw()
sentences = sent_tokenize(genesis_corpus)
print("Number of senteces found in the genesis_corpus: ", len(sentences))
# Removing punctuation and stop words and doing stemming:

stop_words =  set(stopwords.words('english'))
```

Mrunal Labhe, A1 , 12

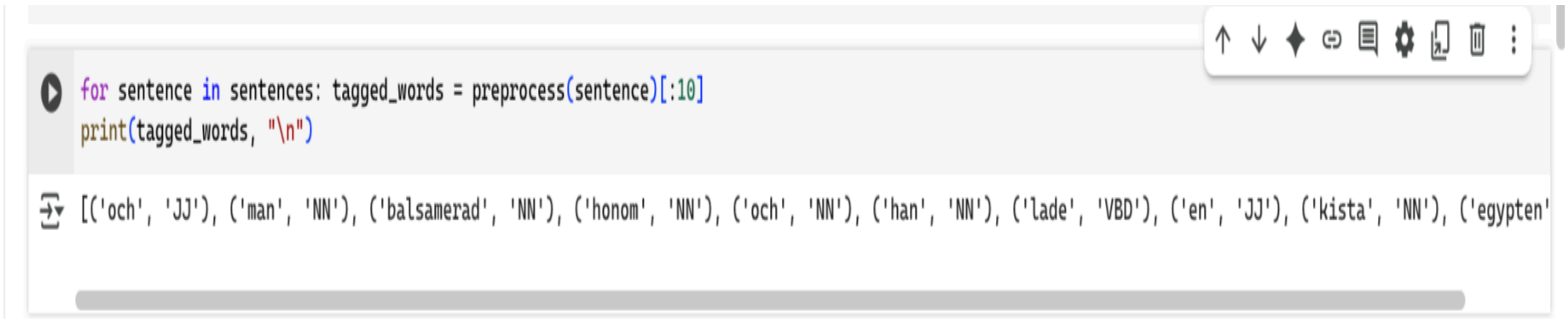
```
stemmer = SnowballStemmer('english')
```
# preprocessing
```
def preprocess(sentence):
 words = word_tokenize(sentence)
 words = [word for word in words  if word.lower() not in stop_words]
 words = [word for word in words if word not in string.punctuation]
 words= [stemmer.stem(word) for word in words]
 tagged_words = pos_tag(words)
 return tagged_words
#calling the function
for sentence in sentences: tagged_words = preprocess(sentence)[:10]
print(tagged_words, "\n")

import spacy.cli
spacy.cli.download("en_core_web_sm")

import spacy
from spacy import displacy
nlp = spacy.load('en_core_web_sm')
text =  "Teaching NLP requires understanding both syntax rules and
semantic analysis"
doc = nlp(text)
displacy.render(doc ,style='dep')
```

Output screenshots:

1.1 Printing the tagged words:



```
for sentence in sentences: tagged_words = preprocess(sentence)[:10]
print(tagged_words, "\n")
```

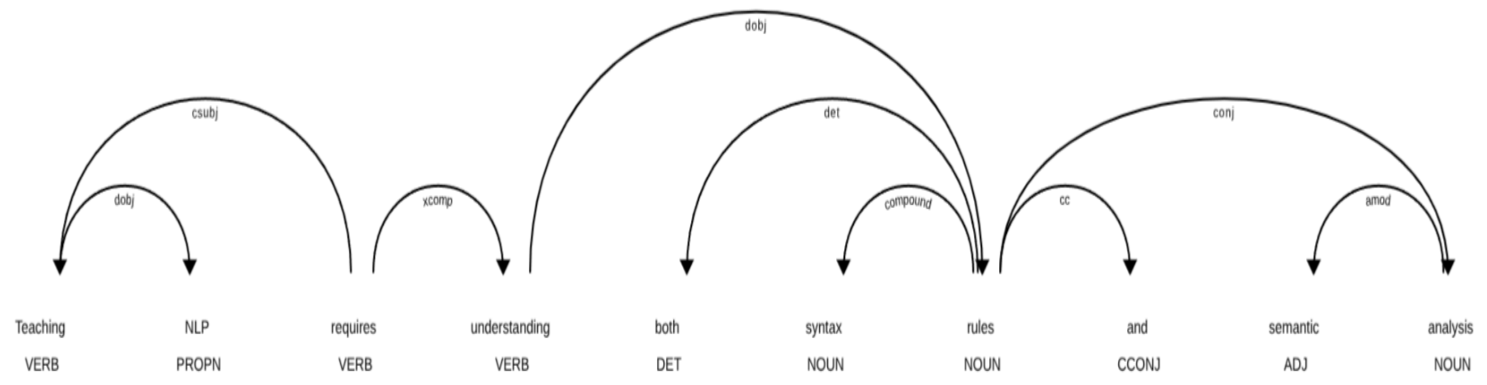
```
[('och', 'JJ'), ('man', 'NN'), ('balsamerad', 'NN'), ('honom', 'NN'), ('och', 'NN'), ('han', 'NN'), ('lade', 'VBD'), ('en', 'JJ'), ('kista', 'NN'), ('egypten'
```

The tagged words are printed after the removal of stop words , all the punctuation and stemming.

2

\1.2 Display figure of words and relationships

```
[ ] displacy.render(doc ,style='dep')
```



Dataset or Resources Used:

- NLTK Genesis Corpus: A publicly available text dataset from the Natural Language Toolkit.
- NLTK Library: Python package for natural language processing.

Procedure Followed:

1. Load and display text from the Genesis corpus with NLTK.
2. Tokenize the text into a list of words.
3. Remove punctuation and stopwords.
4. Apply stemming to tokens.
5. Perform POS tagging.
6. Visualize grammatical structure using a dependency parser.
7. Provide annotated code and explanations.

Date of Performance: 8th August 2025

Viva Questions with Answers

1. **What is the purpose of removing stopwords in NLP?**
   - *Answer:* Stopwords are common words (like "the", "is") that do not contribute significant meaning to text analysis. Removing them reduces noise and improves processing efficiency.
2. **How does stemming differ from lemmatization?**
   - *Answer:* Stemming crudely removes affixes to reduce words to a root form, sometimes producing non-words (like "run" from "running"). Lemmatization finds the dictionary root of a word, ensuring valid words.
3. **What is POS tagging and why is it important?**
   - *Answer:* POS tagging assigns grammatical roles (noun, verb, etc.) to each word, enabling syntactic and semantic understanding, and supports tasks like parsing and named entity recognition.
4. **Explain the purpose of a dependency parser.**
   - *Answer:* A dependency parser maps out grammatical relationships between words, showing how they connect. This helps understand sentence structure, such as which word is the subject or object.
5. **What does the NLTK Genesis corpus contain?**
   - *Answer:* The Genesis corpus contains the text of the biblical Book of Genesis in several languages, used for linguistic and computational language processing studies.