Edge Intelligence Lab

P Sushma

25MML0050

Dataset Link:

https://www.gutenberg.org/cache/epub/77428/pg77428.txt

Introduction:

The code executed demonstrates several fundamental Natural Language Processing (NLP) techniques applied to a literary text. The process begins with downloading a novel from Project Gutenberg, followed by cleaning and preparing the text for analysis. Essential preprocessing steps such as tokenization, stop-word removal, and stemming are then performed to structure the data effectively.

After preprocessing, various linguistic insights are extracted, including frequency distributions of words, part-of-speech tagging, named entity recognition, and the identification of common phrases using N-grams. Together, these steps provide a comprehensive overview of the text's linguistic patterns and overall structure.

Approach:

Resolution of Error Exceptions in NLTK

During the execution of various NLP tasks in the notebook, several Error exceptions were encountered due to missing NLTK resources. These issues were systematically identified and resolved:

1. Sentence Tokenization Error

Issue:
While performing sentence tokenization using nltk.sent_tokenize (cell ID: *LXLY-NbId7J*), an error occurred indicating that the resource punkt _tab was not found.

```
[4] 0s
import re
text_lower = clean_text.lower()
text_no_punct = re.sub(r"[^a-z0-9\s]", " ", text_lower)
text_clean = re.sub(r"\s+", " ", text_no_punct).strip()

# Tokenization
from nltk.tokenize import sent_tokenize, word_tokenize
sentences = sent_tokenize(clean_text)
words = word_tokenize(text_clean)

print(f"Sentences: {len(sentences)}")
print(f"Words: {len(words)}")
```

```
---------------------------------------------------------------------------
LookupError                               Traceback (most recent call last)
/tmp/ipython-input-2435454435.py in <cell line: 0>()
      6 # Tokenization
      7 from nltk.tokenize import sent_tokenize, word_tokenize
----> 8 sentences = sent_tokenize(clean_text)
      9 words = word_tokenize(text_clean)
     10

                                 4 frames
/usr/local/lib/python3.12/dist-packages/nltk/data.py in find(resource_name, paths)
    577     sep = "*" * 70
    578     resource_not_found = f"\n{sep}\n{msg}\n{sep}\n"
--> 579     raise LookupError(resource_not_found)
    580
    581

LookupError:
**********************************************************************
  Resource punkt_tab not found.
  Please use the NLTK Downloader to obtain the resource:

  >>> import nltk
  >>> nltk.download('punkt_tab')

  For more information see: https://www.nltk.org/data.html

  Attempted to load tokenizers/punkt_tab/english/

  Searched in:
    - '/root/nltk_data'
    - '/usr/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/local/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/local/lib/nltk_data'
**********************************************************************
```

Solution:

The issue was resolved by downloading the required tokenizer model using:

nltk.download('punkt_tab')

This ensured the sentence tokeniser functioned correctly.

```
[6] 0s
import re
import nltk
nltk.download('punkt_tab') # Download the missing resource

text_lower = clean_text.lower()
text_no_punct = re.sub(r"[^a-z0-9\s]", " ", text_lower)
text_clean = re.sub(r"\s+", " ", text_no_punct).strip()

# Tokenization
from nltk.tokenize import sent_tokenize, word_tokenize
sentences = sent_tokenize(clean_text)
words = word_tokenize(text_clean)

print(f"Sentences: {len(sentences)}")
print(f"Words: {len(words)}")
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
Sentences: 6854
Words: 80268
```

2. Part-of-Speech (POS) Tagging Error

Issue:

During POS tagging with nltk.pos_tag (cell ID: Ka1YUQdlK74U), an error was raised for the missing resource **averaged_perceptron_tagger_eng**.

```
[12]
import nltk
nltk.download('averaged_perceptron_tagger_eng') # Download the missing resource

sample_tokens = words_no_sw[:500]
pos_tags = nltk.pos_tag(sample_tokens)
print("\nSample POS tags (first 25):")
for token, tag in pos_tags[:25]:
    print(f"{token}: {tag}")
```

```
---------------------------------------------------------------------------
LookupError                               Traceback (most recent call last)
/tmp/ipython-input-2953455207.py in <cell line: 0>()
      1 sample_tokens = words_no_sw[:500]
----> 2 pos_tags = nltk.pos_tag(sample_tokens)
      3 print("\nSample POS tags (first 25):")
      4 for token, tag in pos_tags[:25]:
      5     print(f"{token}: {tag}")

                                 4 frames
/usr/local/lib/python3.12/dist-packages/nltk/data.py in find(resource_name, paths)
    577     sep = "*" * 70
    578     resource_not_found = f"\n{sep}\n{msg}\n{sep}\n"
--> 579     raise LookupError(resource_not_found)
    580
    581

LookupError:
**********************************************************************
  Resource averaged_perceptron_tagger_eng not found.
  Please use the NLTK Downloader to obtain the resource:

  >>> import nltk
  >>> nltk.download('averaged_perceptron_tagger_eng')

  For more information see: https://www.nltk.org/data.html

  Attempted to load taggers/averaged_perceptron_tagger_eng/

  Searched in:
    - '/root/nltk_data'
    - '/usr/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/share/nltk_data'
    - '/usr/local/share/nltk_data'
    - '/usr/lib/nltk_data'
    - '/usr/local/lib/nltk_data'
**********************************************************************
```

Solution:

The required POS tagger was downloaded by adding:

nltk.download('averaged_perceptron_tagger_eng')

This allowed POS tagging to execute without further errors.

```
[13]
✓ 0s
import nltk
nltk.download('averaged_perceptron_tagger_eng') # Download the missing resource

sample_tokens = words_no_sw[:500]
pos_tags = nltk.pos_tag(sample_tokens)
print("\nSample POS tags (first 25):")
for token, tag in pos_tags[:25]:
    print(f"{token}: {tag}")
```

```
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.

Sample POS tags (first 25):
start: NN
project: NN
gutenberg: NN
ebook: VB
black: JJ
parrot: NN
black: JJ
parrot: NN
tale: NN
golden: JJ
chersonese: JJ
harry: NN
hervey: NN
author: NN
caravans: NNS
night: NN
etc: VBP
perceive: JJ
grace: NN
romance: NN
man: NN
exalted: VBD
animals: NNS
james: NNS
branch: VBP
```

3. Named Entity Recognition (NER) Error

Issue:
While performing Named Entity Recognition using nltk.ne_chunk (cell ID: TD4MFYl_LDSf), an error was encountered due to the absence of the **maxent_ne_chunker_tab** resource.

Solution:
The error was resolved by downloading the necessary NER chunker using:

nltk.download('maxent_ne_chunker_tab')

This provided the required models for successful named entity recognition.

```
sample_sents = sentences[:10]
tokenized_sents = [word_tokenize(s) for s in sample_sents]
pos_tagged_sents = [nltk.pos_tag(ts) for ts in tokenized_sents]
ner_chunks = [nltk.ne_chunk(tags) for tags in pos_tagged_sents]

print("\nNamed entities (sample):")
for tree in ner_chunks:
    for subtree in tree:
        if hasattr(subtree, "label") and subtree.label() in ("PERSON", "ORGANIZATION", "GPE"):
            entity = " ".join(token for token, _ in subtree.leaves())
            print(f"{subtree.label()}: {entity}")
```

```
Named entities (sample):
ORGANIZATION: THE
ORGANIZATION: PROJECT
ORGANIZATION: BLACK
ORGANIZATION: PARROT
ORGANIZATION: THE
ORGANIZATION: BLACK
ORGANIZATION: PARROT _A Tale
ORGANIZATION: Golden Chersonese_
ORGANIZATION: HARRY
ORGANIZATION: ETC
GPE: Romance
ORGANIZATION: JAMES
ORGANIZATION: BRANCH
ORGANIZATION: CABELL
ORGANIZATION: THE
ORGANIZATION: CENTURY
ORGANIZATION: BEDELL
ORGANIZATION: Khmers
PERSON: Manipur
PERSON: Arakan
ORGANIZATION: Lake
PERSON: Tonle Sap
GPE: Angkor
ORGANIZATION: Tevadas
PERSON: Naga
ORGANIZATION: MAN
ORGANIZATION: FROM
ORGANIZATION: BLUE
PERSON: _Cambodia_ V
ORGANIZATION: CONQUEST
ORGANIZATION: DREAM
ORGANIZATION: MALAY
ORGANIZATION: HOUSE
ORGANIZATION: BARABBAS
ORGANIZATION: BLACK
ORGANIZATION: PARROT
ORGANIZATION: MAN
ORGANIZATION: FROM
ORGANIZATION: Pacific
GPE: Nowhere
```

Summary:

- Collected a literary text from Project Gutenberg and cleaned it by removing unwanted sections and standardizing the formatting.
- Preprocessed the text through lowercasing, punctuation removal, and sentence/word tokenization.
- Refined the vocabulary by removing stopwords and applying stemming.
- Analysed the text using frequency distributions, POS tagging, and Named Entity Recognition to understand word usage, grammar roles, and key entities.

- Explored patterns through N-gram analysis and concordance searches to study common phrases and context.
- Resolved several NLTK Error issues by downloading the required resources, ensuring all NLP tasks ran smoothly.

Conclusion:

Overall, the notebook successfully demonstrates a complete NLP workflow applied to a literary text from data acquisition and cleaning to detailed linguistic analysis. Through systematic preprocessing and the use of various NLP techniques such as frequency analysis, POS tagging, NER, and N-grams, the text was explored from multiple linguistic perspectives. The resolution of required NLTK resource errors ensured smooth execution and reliability throughout the process. This workflow provides a strong foundation for understanding text structure, extracting meaningful information, and preparing data for more advanced NLP or machine learning tasks.