Navadeep Munugoti

2403A52015

AIAI 02

```
import nltk
import spacy

text = "The old clock on the mantelpiece ticked rhythmically, its
steady cadence a comforting backdrop to the quiet room. Dust motes
danced in the lone shaft of sunlight piercing through the heavy velvet
curtains, illuminating forgotten corners and the well-worn pages of an
open book. Outside, the world hummed with its usual cacophony, but
within these walls, time seemed to slow, offering a brief respite from
the hurried pace of modern life."

nltk.download('punkt') # Download the punkt tokenizer models
nltk.download('punkt_tab') # Download punkt_tab as suggested by the
error
tokens = nltk.word_tokenize(text)
display(tokens)
```

```
[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]   Unzipping tokenizers/punkt_tab.zip.
```

```
['The',
 'old',
 'clock',
 'on',
 'the',
 'mantelpiece',
 'ticked',
 'rhythmically',
 ',',
 'its',
 'steady',
 'cadence',
 'a',
 'comforting',
 'backdrop',
 'to',
 'the',
 'quiet',
 'room',
 '.',
 'Dust',
 'motes',
```

```
 'danced',
 'in',
 'the',
 'lone',
 'shaft',
 'of',
 'sunlight',
 'piercing',
 'through',
 'the',
 'heavy',
 'velvet',
 'curtains',
 ',',
 'illuminating',
 'forgotten',
 'corners',
 'and',
 'the',
 'well-worn',
 'pages',
 'of',
 'an',
 'open',
 'book',
 '.',
 'Outside',
 ',',
 'the',
 'world',
 'hummed',
 'with',
 'its',
 'usual',
 'cacophony',
 ',',
 'but',
 'within',
 'these',
 'walls',
 ',',
 'time',
 'seemed',
 'to',
 'slow',
 ',',
 'offering',
 'a',
 'brief',
```

```
 'respite',
 'from',
 'the',
 'hurried',
 'pace',
 'of',
 'modern',
 'life',
 '.']
```

```
nltk.download('averaged_perceptron_tagger_eng') # Download the
specific English POS tagger data
pos_tags = nltk.pos_tag(tokens)
display(pos_tags)
```

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

```
[('The', 'DT'),
 ('old', 'JJ'),
 ('clock', 'NN'),
 ('on', 'IN'),
 ('the', 'DT'),
 ('mantelpiece', 'NN'),
 ('ticked', 'VBD'),
 ('rhythmically', 'RB'),
 (',', ','),
 ('its', 'PRP$'),
 ('steady', 'JJ'),
 ('cadence', 'NN'),
 ('a', 'DT'),
 ('comforting', 'VBG'),
 ('backdrop', 'NN'),
 ('to', 'TO'),
 ('the', 'DT'),
 ('quiet', 'JJ'),
 ('room', 'NN'),
 ('.', '.'),
 ('Dust', 'NNP'),
 ('motes', 'NNS'),
 ('danced', 'VBD'),
 ('in', 'IN'),
 ('the', 'DT'),
 ('lone', 'NN'),
 ('shaft', 'NN'),
 ('of', 'IN'),
 ('sunlight', 'NN'),
```

```
('piercing', 'VBG'),
('through', 'IN'),
('the', 'DT'),
('heavy', 'JJ'),
('velvet', 'NN'),
('curtains', 'NNS'),
(',', ','),
('illuminating', 'VBG'),
('forgotten', 'JJ'),
('corners', 'NNS'),
('and', 'CC'),
('the', 'DT'),
('well-worn', 'JJ'),
('pages', 'NNS'),
('of', 'IN'),
('an', 'DT'),
('open', 'JJ'),
('book', 'NN'),
('.', '.'),
('Outside', 'NNP'),
(',', ','),
('the', 'DT'),
('world', 'NN'),
('hummed', 'VBD'),
('with', 'IN'),
('its', 'PRP$'),
('usual', 'JJ'),
('cacophony', 'NN'),
(',', ','),
('but', 'CC'),
('within', 'IN'),
('these', 'DT'),
('walls', 'NNS'),
(',', ','),
('time', 'NN'),
('seemed', 'VBD'),
('to', 'TO'),
('slow', 'VB'),
(',', ','),
('offering', 'VBG'),
('a', 'DT'),
('brief', 'JJ'),
('respite', 'NN'),
('from', 'IN'),
('the', 'DT'),
('hurried', 'JJ'),
('pace', 'NN'),
('of', 'IN'),
('modern', 'JJ'),
```

```
 ('life', 'NN'),
 ('.', '.')]
```

```
import spacy

try:
    nlp = spacy.load('en_core_web_sm')
except OSError:
    print('Downloading spaCy English model (en_core_web_sm)...')
    !python -m spacy download en_core_web_sm
    nlp = spacy.load('en_core_web_sm')

doc = nlp(text)

spacy_pos_tags = []
for token in doc:
    spacy_pos_tags.append((token.text, token.pos_))

display(spacy_pos_tags)
```

```
[('The', 'DET'),
 ('old', 'ADJ'),
 ('clock', 'NOUN'),
 ('on', 'ADP'),
 ('the', 'DET'),
 ('mantelpiece', 'NOUN'),
 ('ticked', 'VERB'),
 ('rhythmically', 'PROPN'),
 (',', 'PUNCT'),
 ('its', 'PRON'),
 ('steady', 'ADJ'),
 ('cadence', 'NOUN'),
 ('a', 'DET'),
 ('comforting', 'VERB'),
 ('backdrop', 'NOUN'),
 ('to', 'ADP'),
 ('the', 'DET'),
 ('quiet', 'ADJ'),
 ('room', 'NOUN'),
 ('.', 'PUNCT'),
 ('Dust', 'NOUN'),
 ('motes', 'NOUN'),
 ('danced', 'VERB'),
 ('in', 'ADP'),
 ('the', 'DET'),
 ('lone', 'ADJ'),
 ('shaft', 'NOUN'),
 ('of', 'ADP'),
```

```
('sunlight', 'NOUN'),
('piercing', 'VERB'),
('through', 'ADP'),
('the', 'DET'),
('heavy', 'ADJ'),
('velvet', 'NOUN'),
('curtains', 'NOUN'),
(',', 'PUNCT'),
('illuminating', 'VERB'),
('forgotten', 'VERB'),
('corners', 'NOUN'),
('and', 'CCONJ'),
('the', 'DET'),
('well', 'ADV'),
('-', 'PUNCT'),
('worn', 'VERB'),
('pages', 'NOUN'),
('of', 'ADP'),
('an', 'DET'),
('open', 'ADJ'),
('book', 'NOUN'),
('.', 'PUNCT'),
('Outside', 'ADV'),
(',', 'PUNCT'),
('the', 'DET'),
('world', 'NOUN'),
('hummed', 'VERB'),
('with', 'ADP'),
('its', 'PRON'),
('usual', 'ADJ'),
('cacophony', 'NOUN'),
(',', 'PUNCT'),
('but', 'CCONJ'),
('within', 'ADP'),
('these', 'DET'),
('walls', 'NOUN'),
(',', 'PUNCT'),
('time', 'NOUN'),
('seemed', 'VERB'),
('to', 'PART'),
('slow', 'VERB'),
(',', 'PUNCT'),
('offering', 'VERB'),
('a', 'DET'),
('brief', 'ADJ'),
('respite', 'NOUN'),
('from', 'ADP'),
('the', 'DET'),
('hurried', 'ADJ'),
```

```
 ('pace', 'NOUN'),
 ('of', 'ADP'),
 ('modern', 'ADJ'),
 ('life', 'NOUN'),
 ('.', 'PUNCT')]
```

```
unique_nltk_pos_tags = set()
for _, tag in pos_tags:
    unique_nltk_pos_tags.add(tag)
display(unique_nltk_pos_tags)
```

```
{',',
 '.',
 'CC',
 'DT',
 'IN',
 'JJ',
 'NN',
 'NNP',
 'NNS',
 'PRP$',
 'RB',
 'TO',
 'VB',
 'VBD',
 'VBG'}
```

```
unique_spacy_pos_tags = set()
for _, tag in spacy_pos_tags:
    unique_spacy_pos_tags.add(tag)
display(unique_spacy_pos_tags)
```

```
{'ADJ',
 'ADP',
 'ADV',
 'CCONJ',
 'DET',
 'NOUN',
 'PART',
 'PRON',
 'PROPN',
 'PUNCT',
 'VERB'}
```

```
common_pos_tags =
unique_nltk_pos_tags.intersection(unique_spacy_pos_tags)
nltk_only_pos_tags =
unique_nltk_pos_tags.difference(unique_spacy_pos_tags)
spacy_only_pos_tags =
```

```
unique_spacy_pos_tags.difference(unique_nltk_pos_tags)

print("Common POS Tags:")
display(common_pos_tags)

print("NLTK-only POS Tags:")
display(nltk_only_pos_tags)

print("spaCy-only POS Tags:")
display(spacy_only_pos_tags)
```

```
Common POS Tags:

set()

NLTK-only POS Tags:

{',',
 '.',
 'CC',
 'DT',
 'IN',
 'JJ',
 'NN',
 'NNP',
 'NNS',
 'PRP$',
 'RB',
 'TO',
 'VB',
 'VBD',
 'VBG'}

spaCy-only POS Tags:

{'ADJ',
 'ADP',
 'ADV',
 'CCONJ',
 'DET',
 'NOUN',
 'PART',
 'PRON',
 'PROPN',
 'PUNCT',
 'VERB'}
```

Identify Academic Concepts (Nouns) and Arguments (Verbs) using NLTK

```
nltk_academic_nouns = set()
nltk_argument_verbs = set()

# NLTK noun tags (academic concepts)
nltk_noun_tags = {'NN', 'NNS', 'NNP', 'NNPS'}
# NLTK verb tags (arguments/actions)
nltk_verb_tags = {'VB', 'VBD', 'VBG', 'VBN', 'VBP', 'VBZ'}

for word, tag in pos_tags:
    if tag in nltk_noun_tags:
        nltk_academic_nouns.add(word.lower())
    elif tag in nltk_verb_tags:
        nltk_argument_verbs.add(word.lower())

print("NLTK Academic Concepts (Nouns):")
display(sorted(list(nltk_academic_nouns)))

print("NLTK Arguments (Verbs):")
display(sorted(list(nltk_argument_verbs)))
```

```
NLTK Academic Concepts (Nouns):

['backdrop',
 'book',
 'cacophony',
 'cadence',
 'clock',
 'corners',
 'curtains',
 'dust',
 'life',
 'lone',
 'mantelpiece',
 'motes',
 'outside',
 'pace',
 'pages',
 'respite',
 'room',
 'shaft',
 'sunlight',
 'time',
 'velvet',
 'walls',
 'world']

NLTK Arguments (Verbs):
```

```
['comforting',
 'danced',
 'hummed',
 'illuminating',
 'offering',
 'piercing',
 'seemed',
 'slow',
 'ticked']
```

Identify Academic Concepts (Nouns) and Arguments (Verbs) using spaCy

```
spacy_academic_nouns = set()
spacy_argument_verbs = set()

# spaCy noun tags (academic concepts)
spacy_noun_tags = {'NOUN', 'PROPN'}
# spaCy verb tags (arguments/actions)
spacy_verb_tags = {'VERB'}

for word, tag in spacy_pos_tags:
    if tag in spacy_noun_tags:
        spacy_academic_nouns.add(word.lower())
    elif tag in spacy_verb_tags:
        spacy_argument_verbs.add(word.lower())

print("spaCy Academic Concepts (Nouns):")
display(sorted(list(spacy_academic_nouns)))

print("spaCy Arguments (Verbs):")
display(sorted(list(spacy_argument_verbs)))
```

```
spaCy Academic Concepts (Nouns):

['backdrop',
 'book',
 'cacophony',
 'cadence',
 'clock',
 'corners',
 'curtains',
 'dust',
 'life',
 'mantelpiece',
 'motes',
 'pace',
 'pages',
 'respite',
```

```
 'rhythmically',
 'room',
 'shaft',
 'sunlight',
 'time',
 'velvet',
 'walls',
 'world']

spaCy Arguments (Verbs):

['comforting',
 'danced',
 'forgotten',
 'hummed',
 'illuminating',
 'offering',
 'piercing',
 'seemed',
 'slow',
 'ticked',
 'worn']
```

```
from collections import Counter

# Calculate NLTK Noun Frequencies
nltk_noun_frequencies = Counter()
for word, tag in pos_tags:
    if tag in nltk_noun_tags:
        nltk_noun_frequencies[word.lower()] += 1

print("NLTK Academic Noun Frequencies:")
display(nltk_noun_frequencies.most_common())

# Calculate NLTK Verb Frequencies
nltk_verb_frequencies = Counter()
for word, tag in pos_tags:
    if tag in nltk_verb_tags:
        nltk_verb_frequencies[word.lower()] += 1

print("NLTK Argument Verb Frequencies:")
display(nltk_verb_frequencies.most_common())
```

```
NLTK Academic Noun Frequencies:

[('clock', 1),
 ('mantelpiece', 1),
 ('cadence', 1),
 ('backdrop', 1),
 ('room', 1),
 ('dust', 1),
```

```
 ('motes', 1),
 ('lone', 1),
 ('shaft', 1),
 ('sunlight', 1),
 ('velvet', 1),
 ('curtains', 1),
 ('corners', 1),
 ('pages', 1),
 ('book', 1),
 ('outside', 1),
 ('world', 1),
 ('cacophony', 1),
 ('walls', 1),
 ('time', 1),
 ('respite', 1),
 ('pace', 1),
 ('life', 1)]

NLTK Argument Verb Frequencies:

[('ticked', 1),
 ('comforting', 1),
 ('danced', 1),
 ('piercing', 1),
 ('illuminating', 1),
 ('hummed', 1),
 ('seemed', 1),
 ('slow', 1),
 ('offering', 1)]
```

```
from collections import Counter

# Calculate spaCy Noun Frequencies
spacy_noun_frequencies = Counter()
for token in doc:
    if token.pos_ in spacy_noun_tags:
        spacy_noun_frequencies[token.text.lower()] += 1

print("spaCy Academic Noun Frequencies:")
display(spacy_noun_frequencies.most_common())

# Calculate spaCy Verb Frequencies
spacy_verb_frequencies = Counter()
for token in doc:
    if token.pos_ in spacy_verb_tags:
        spacy_verb_frequencies[token.text.lower()] += 1

print("spaCy Argument Verb Frequencies:")
display(spacy_verb_frequencies.most_common())
```

```
spaCy Academic Noun Frequencies:
```

```
[('clock', 1),
 ('mantelpiece', 1),
 ('rhythmically', 1),
 ('cadence', 1),
 ('backdrop', 1),
 ('room', 1),
 ('dust', 1),
 ('motes', 1),
 ('shaft', 1),
 ('sunlight', 1),
 ('velvet', 1),
 ('curtains', 1),
 ('corners', 1),
 ('pages', 1),
 ('book', 1),
 ('world', 1),
 ('cacophony', 1),
 ('walls', 1),
 ('time', 1),
 ('respite', 1),
 ('pace', 1),
 ('life', 1)]

spaCy Argument Verb Frequencies:

[('ticked', 1),
 ('comforting', 1),
 ('danced', 1),
 ('piercing', 1),
 ('illuminating', 1),
 ('forgotten', 1),
 ('worn', 1),
 ('hummed', 1),
 ('seemed', 1),
 ('slow', 1),
 ('offering', 1)]

import pandas as pd

# Convert NLTK Noun Frequencies to DataFrame
nltk_nouns_df = pd.DataFrame(nltk_noun_frequencies.most_common(),
columns=['Noun', 'Frequency'])
print("NLTK Academic Noun Frequencies (DataFrame):")
display(nltk_nouns_df)

# Convert NLTK Verb Frequencies to DataFrame
nltk_verbs_df = pd.DataFrame(nltk_verb_frequencies.most_common(),
columns=['Verb', 'Frequency'])
print("NLTK Argument Verb Frequencies (DataFrame):")
display(nltk_verbs_df)
```

```
NLTK Academic Noun Frequencies (DataFrame):
```


```
{"summary":"{\n  \"name\": \"nltk_nouns_df\",\n  \"rows\": 23,\n
\"fields\": [\n    {\n      \"column\": \"Noun\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 23,\n        \"samples\": [\n
\"outside\",\n          \"sunlight\",\n          \"clock\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Frequency\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0,\n        \"min\": 1,\n
\"max\": 1,\n        \"num_unique_values\": 1,\n        \"samples\":
[\n          1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"nltk_nouns_df"}
```


```
NLTK Argument Verb Frequencies (DataFrame):
```


```
{"summary":"{\n  \"name\": \"nltk_verbs_df\",\n  \"rows\": 9,\n
\"fields\": [\n    {\n      \"column\": \"Verb\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 9,\n        \"samples\": [\n
\"slow\",\n          \"comforting\",\n          \"hummed\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Frequency\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0,\n        \"min\": 1,\n
\"max\": 1,\n        \"num_unique_values\": 1,\n        \"samples\":
[\n          1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"nltk_verbs_df"}
```


```
import pandas as pd

# Convert spaCy Noun Frequencies to DataFrame
spacy_nouns_df = pd.DataFrame(spacy_noun_frequencies.most_common(),
columns=['Noun', 'Frequency'])
print("spaCy Academic Noun Frequencies (DataFrame):")
display(spacy_nouns_df)

# Convert spaCy Verb Frequencies to DataFrame
spacy_verbs_df = pd.DataFrame(spacy_verb_frequencies.most_common(),
columns=['Verb', 'Frequency'])
print("spaCy Argument Verb Frequencies (DataFrame):")
display(spacy_verbs_df)
```

```
spaCy Academic Noun Frequencies (DataFrame):
```


```
{"summary":"{\n  \"name\": \"spacy_nouns_df\",\n  \"rows\": 22,\n
\"fields\": [\n    {\n      \"column\": \"Noun\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
```

\"num_unique_values\": 22,\n   \"samples\": [\n
\"clock\",\n   \"pages\",\n   \"shaft\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Frequency\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0,\n   \"min\": 1,\n   \"max\": 1,\n
\"num_unique_values\": 1,\n   \"samples\": [\n   1\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   }\n]\n}","type":"dataframe","variable_name":"spacy_nouns_df"}

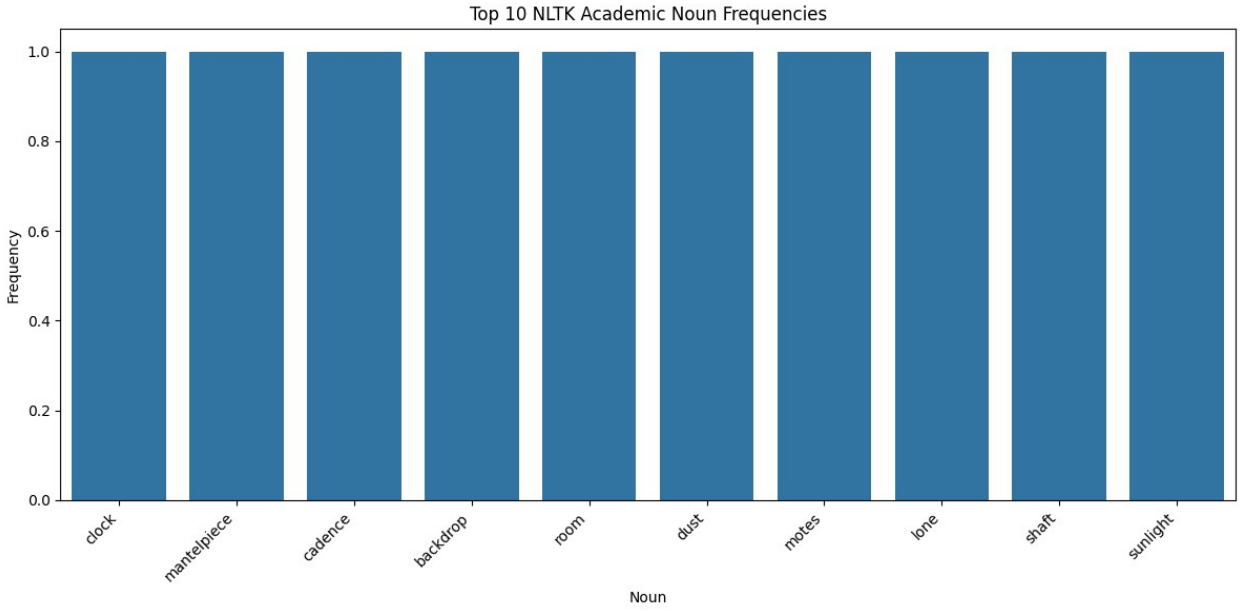
spaCy Argument Verb Frequencies (DataFrame):

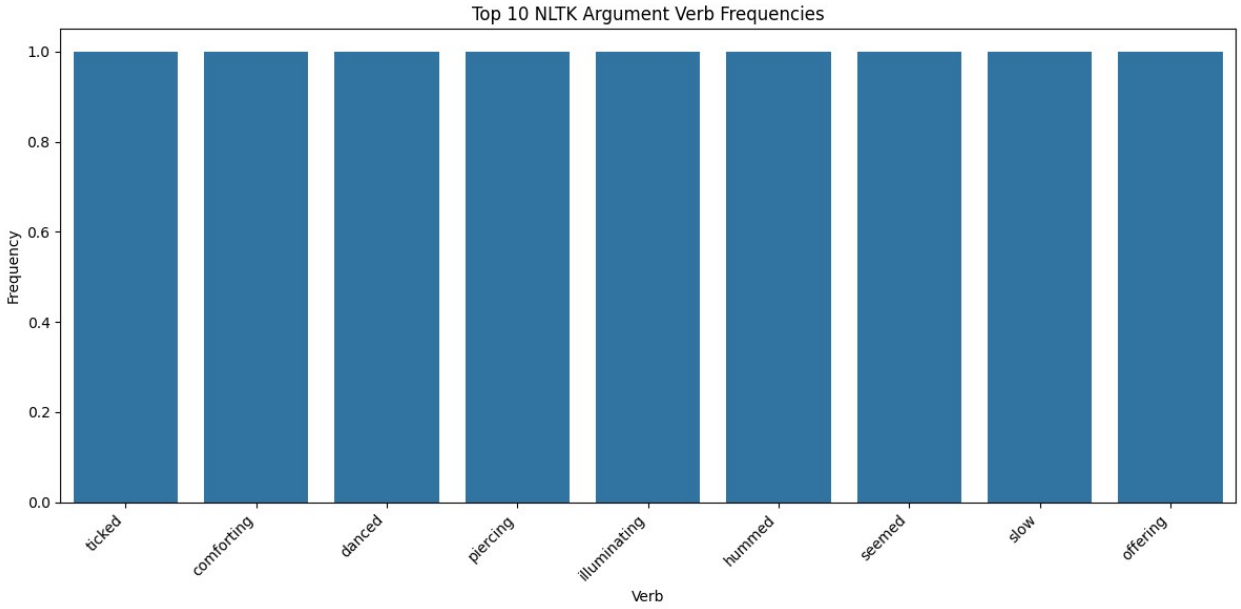
{"summary":"{\n \"name\": \"spacy_verbs_df\",\n \"rows\": 11,\n
\"fields\": [\n   {\n   \"column\": \"Verb\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 11,\n   \"samples\": [\n
\"forgotten\",\n   \"ticked\",\n   \"slow\"\n   ],\
n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Frequency\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0,\n   \"min\": 1,\n   \"max\": 1,\n
\"num_unique_values\": 1,\n   \"samples\": [\n   1\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   }\n]\n}","type":"dataframe","variable_name":"spacy_verbs_df"}

```
import matplotlib.pyplot as plt
import seaborn as sns

# Visualize NLTK Academic Noun Frequencies
plt.figure(figsize=(12, 6))
sns.barplot(x='Noun', y='Frequency', data=nltk_nouns_df.head(10))
plt.title('Top 10 NLTK Academic Noun Frequencies')
plt.xlabel('Noun')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# Visualize NLTK Argument Verb Frequencies
plt.figure(figsize=(12, 6))
sns.barplot(x='Verb', y='Frequency', data=nltk_verbs_df.head(10))
plt.title('Top 10 NLTK Argument Verb Frequencies')
plt.xlabel('Verb')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```






```python
import matplotlib.pyplot as plt
import seaborn as sns

# Visualize spaCy Academic Noun Frequencies
plt.figure(figsize=(12, 6))
sns.barplot(x='Noun', y='Frequency', data=spacy_nouns_df.head(10))
plt.title('Top 10 spaCy Academic Noun Frequencies')
plt.xlabel('Noun')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

```
# Visualize spaCy Argument Verb Frequencies
plt.figure(figsize=(12, 6))
sns.barplot(x='Verb', y='Frequency', data=spacy_verbs_df.head(10))
plt.title('Top 10 spaCy Argument Verb Frequencies')
plt.xlabel('Verb')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```