

```
import sys
print(sys.version)
```

3.12.12 (main, Oct 10 2025, 08:52:57) [GCC 11.4.0]

```
!pip install nltk
```

Requirement already satisfied: nltk in /usr/local/lib/python3.12/dist-packages (3.9.1)
 Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages (from nltk) (8.3.1)
 Requirement already satisfied: joblib in /usr/local/lib/python3.12/dist-packages (from nltk) (1.5.3)
 Requirement already satisfied: regex<=2021.8.3 in /usr/local/lib/python3.12/dist-packages (from nltk) (2025.11.3)
 Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (from nltk) (4.67.1)

```
!pip install spacy
```

Requirement already satisfied: spacy in /usr/local/lib/python3.12/dist-packages (3.8.11)
 Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.12/dist-packages (from spacy) (3.0.12)
 Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (1.0.5)
 Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (1.0.15)
 Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.0.13)
 Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (3.0.12)
 Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/python3.12/dist-packages (from spacy) (8.3.10)
 Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.12/dist-packages (from spacy) (1.1.3)
 Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.5.2)
 Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.0.10)
 Requirement already satisfied: weasel<0.5.0,>=0.4.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (0.4.3)
 Requirement already satisfied: typer-slim<1.0.0,>=0.3.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (0.21.1)
 Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (4.67.1)
 Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.0.2)
 Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.32.4)
 Requirement already satisfied: pydantic!=1.8,!1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.12/dist-packages (from spacy)
 Requirement already satisfied: Jinja2 in /usr/local/lib/python3.12/dist-packages (from spacy) (3.1.6)
 Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from spacy) (75.2.0)
 Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (25.0)
 Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!1.8.1,
 Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!1.8.1,
 Requirement already satisfied: typing-extensions>=4.14.1 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!1.8.1,
 Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib/python3.12/dist-packages (from pydantic!=1.8,!1.8.1,
 Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy)
 Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy)
 Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy)
 Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2.13.0->spacy)
 Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/python3.12/dist-packages (from thinc<8.4.0,>=8.3.4->spacy)
 Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.12/dist-packages (from thinc<8.4.0,>=8.3.4->spacy)
 Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.12/dist-packages (from typer-slim<1.0.0,>=0.3.0->spacy)
 Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.12/dist-packages (from weasel<0.5.0,>=0.4.3->spacy)
 Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.12/dist-packages (from weasel<0.5.0,>=0.4.3->spacy)
 Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from Jinja2->spacy) (3.0.3)
 Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from smart-open<8.0.0,>=5.2.1->weasel<0.5.0,>=0.4.3->spacy)

```
import nltk
```

```
nltk.download('punkt')          # Tokenizer
nltk.download('averaged_perceptron_tagger') # POS tagger
nltk.download('stopwords')      # Stopwords (optional)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data] /root/nltk_data...
[nltk_data] Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Unzipping corpora/stopwords.zip.
True
```

```
!python -m spacy download en_core_web_sm
```

Collecting en-core-web-sm==3.8.0
 Downloading https://github.com/explosion/spacy-models/releases/download/en_core_web_sm-3.8.0/en_core_web_sm-3.8.0-py3-none-any.whl 12.8/12.8 MB 100.2 MB/s eta 0:00:00
 ✓ Download and installation successful
 You can now load the package via spacy.load('en_core_web_sm')
 ⚠ Restart to reload dependencies
 If you are in a Jupyter or Colab notebook, you may need to restart Python in order to load all the package's dependencies. You can do this by selecting the 'Restart kernel' or 'Restart runtime' option.

```
import spacy

nlp = spacy.load("en_core_web_sm")
print("spaCy model loaded successfully!")
```

spaCy model loaded successfully!

```
import nltk
import spacy
import pandas as pd
import matplotlib.pyplot as plt
from collections import Counter
```

```
import nltk
from nltk.tokenize import word_tokenize
from nltk import pos_tag

nltk.download('punkt_tab') # Download the missing resource
nltk.download('averaged_perceptron_tagger_eng') # Download the missing resource for English POS tagger
```

```
sample_text = "Academic writing requires clarity and precision."
tokens = word_tokenize(sample_text)
tags = pos_tag(tokens)

print(tags)
```

```
[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_eng to
[nltk_data] /root/nltk_data...
[nltk_data] Unzipping taggers/averaged_perceptron_tagger_eng.zip.
[('Academic', 'NNP'), ('writing', 'NN'), ('requires', 'VBZ'), ('clarity', 'NN'), ('and', 'CC'), ('precision', 'NN'), ('.', 'PUNCT')]
```

```
doc = nlp(sample_text)

for token in doc:
    print(token.text, token.pos_)
```

```
Academic ADJ
writing NOUN
requires VERB
clarity NOUN
and CCONJ
precision NOUN
. PUNCT
```

```
essay_text = """
Academic writing plays a crucial role in higher education.
It allows students to analyze concepts, present arguments,
and demonstrate critical thinking. Research-based writing
requires clarity, logical structure, and proper evidence.
Scholars evaluate theories, compare methodologies, and
discuss results to contribute knowledge to their field.
"""
```

```
from nltk.tokenize import word_tokenize

tokens = word_tokenize(essay_text)
print(tokens)
```

```
['Academic', 'writing', 'plays', 'a', 'crucial', 'role', 'in', 'higher', 'education', '.', 'It', 'allows', 'students', 'to',
```

```
from nltk import pos_tag

nltk_pos_tags = pos_tag(tokens)
print(nltk_pos_tags)
```

```
[('Academic', 'NNP'), ('writing', 'NN'), ('plays', 'VBZ'), ('a', 'DT'), ('crucial', 'JJ'), ('role', 'NN'), ('in', 'IN'), ('It', 'PRP'), ('allows', 'VBZ'), ('students', 'NNS'), ('to', 'TO'), ('discuss', 'VBZ'), ('results', 'NNS'), ('to', 'TO'), ('contribute', 'VB'), ('knowledge', 'NN'), ('to', 'TO'), ('their', 'PRP'), ('field', 'NN'), ('.', 'PUNCT')]
```

```
import spacy

nlp = spacy.load("en_core_web_sm")
```

```
doc = nlp(essay_text)
```

```
spacy_pos_tags = [(token.text, token.pos_) for token in doc]
print(spacy_pos_tags)
```

```
[('\\n', 'SPACE'), ('Academic', 'ADJ'), ('writing', 'NOUN'), ('plays', 'VERB'), ('a', 'DET'), ('crucial', 'ADJ'), ('role', 'N
```

```
import pandas as pd
```

```
# Filter out SPACE tokens from spaCy doc to get a more comparable token list.
# spaCy often tokenizes newlines (\\n) as 'SPACE', which NLTK's word_tokenize typically doesn't.
spacy_filtered_tokens = [token for token in doc if not token.is_space]
spacy_filtered_pos_list = [token.pos_ for token in spacy_filtered_tokens]
```

```
# Determine the minimum length among the NLTK and filtered spaCy token lists
# to ensure all columns in the DataFrame have the same length.
min_len = min(len(nltk_pos_tags), len(spacy_filtered_pos_list))
```

```
comparison_df = pd.DataFrame({
    "Word": [token for token, _ in nltk_pos_tags][:min_len],
    "NLTK_POS": [tag for _, tag in nltk_pos_tags][:min_len],
    "spaCy_POS": spacy_filtered_pos_list[:min_len]
})
```

```
comparison_df.head(10)
```

	Word	NLTK_POS	spaCy_POS
0	Academic	NNP	ADJ
1	writing	NN	NOUN
2	plays	VBZ	VERB
3	a	DT	DET
4	crucial	JJ	ADJ
5	role	NN	NOUN
6	in	IN	ADP
7	higher	JJR	ADJ
8	education	NN	NOUN
9	.	.	PUNCT

Next steps: [Generate code with comparison_df](#) [New interactive sheet](#)

```
nouns = [token.text.lower() for token in doc if token.pos_ == "NOUN"]
verbs = [token.text.lower() for token in doc if token.pos_ == "VERB"]
```

```
print("Nouns:", nouns)
print("Verbs:", verbs)
```

```
Nouns: ['writing', 'role', 'education', 'students', 'concepts', 'arguments', 'thinking', 'research', 'writing', 'clarity', '
Verbs: ['plays', 'allows', 'analyze', 'demonstrate', 'based', 'requires', 'evaluate', 'discuss', 'contribute']
```

```
from collections import Counter
```

```
noun_freq = Counter(nouns)
verb_freq = Counter(verbs)
```

```
print("Noun Frequency:", noun_freq)
print("Verb Frequency:", verb_freq)
```

```
Noun Frequency: Counter({'writing': 2, 'role': 1, 'education': 1, 'students': 1, 'concepts': 1, 'arguments': 1, 'thinking':
Verb Frequency: Counter({'plays': 1, 'allows': 1, 'analyze': 1, 'demonstrate': 1, 'based': 1, 'requires': 1, 'evaluate': 1,
```

```
noun_df = pd.DataFrame(noun_freq.items(), columns=["Noun", "Frequency"])
verb_df = pd.DataFrame(verb_freq.items(), columns=["Verb", "Frequency"])
```

```
noun_df.sort_values(by="Frequency", ascending=False)
```

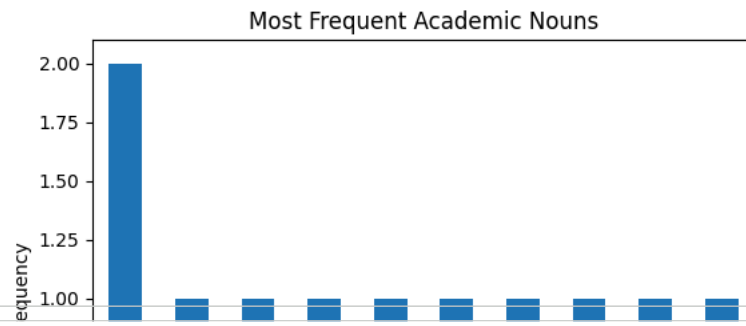
	Noun	Frequency	
0	writing	2	
1	role	1	
2	education	1	
3	students	1	
4	concepts	1	
5	arguments	1	
6	thinking	1	
7	research	1	
8	clarity	1	
9	structure	1	
10	evidence	1	
11	scholars	1	
12	theories	1	
13	compare	1	
14	methodologies	1	
15	results	1	
16	knowledge	1	
17	field	1	

```
verb_df.sort_values(by="Frequency", ascending=False)
```

	Verb	Frequency	
0	plays	1	
1	allows	1	
2	analyze	1	
3	demonstrate	1	
4	based	1	
5	requires	1	
6	evaluate	1	
7	discuss	1	
8	contribute	1	

```
plt.figure()
noun_df.sort_values(by="Frequency", ascending=False).head(10).plot(
    kind="bar",
    x="Noun",
    y="Frequency",
    legend=False
)
plt.title("Most Frequent Academic Nouns")
plt.xlabel("Nouns")
plt.ylabel("Frequency")
plt.show()
```

<Figure size 640x480 with 0 Axes>



```
plt.figure()
verb_df.sort_values(by="Frequency", ascending=False).head(10).plot(
    kind="bar",
    x="Verb",
    y="Frequency",
    legend=False
)
plt.title("Most Frequent Academic Verbs")
plt.xlabel("Verbs")
plt.ylabel("Frequency")
plt.show()
```

<Figure size 640x480 with 0 Axes>

