```
Name:P.Srikar
Roll:2403a52002
```

```
import sys

# Install NLTK
!{sys.executable} -m pip install nltk

# Install spaCy
!{sys.executable} -m pip install spacy

# Download necessary NLTK data (e.g., 'punkt' tokenizer)
import nltk
try:
    nltk.data.find('tokenizers/punkt')
except LookupError:
    nltk.download('punkt')

# Download spaCy English model
# You might need to restart the runtime after installing spaCy for the model download to work correctly
# Alternatively, you can run this command in a separate cell after spaCy is installed
try:
    import spacy
    spacy.load('en_core_web_sm')
except OSError:
    print("Downloading spaCy model 'en_core_web_sm'. This might take a moment...")
    !{sys.executable} -m spacy download en_core_web_sm
    import spacy # Re-import spacy after model download

print("NLTK and spaCy are installed and ready.")
```

```
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.
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (from nltk) (4.67.1)
```

```
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 spa
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.12/dist-packages (from spa
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.12/dist-packages (from spacy
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.0
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (3
Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/python3.12/dist-packages (from spacy) (8.3
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.12/dist-packages (from spacy) (1.
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.12/dist-packages (from spacy) (2.5
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.12/dist-packages (from spacy)
Requirement already satisfied: weasel<0.5.0,>=0.4.2 in /usr/local/lib/python3.12/dist-packages (from spacy) (0.
Requirement already satisfied: typer-slim<1.0.0,>=0.3.0 in /usr/local/lib/python3.12/dist-packages (from spacy)
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.12/dist-packages (from spacy) (4.6
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)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.12/dist-packages
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
Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/python3.12/dist-packages (from pydantic!
Requirement already satisfied: typing-extensions>=4.14.1 in /usr/local/lib/python3.12/dist-packages (from pydan
Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib/python3.12/dist-packages (from pydant
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from reques
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests<3.0.0,>=2
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests<3.0
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests<3.0
Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/python3.12/dist-packages (from thinc<8.4.0,
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.12/dist-packages (from thinc<
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.12/dist-packages (from typer-slim<1.0.0,>
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.12/dist-packages (from weas
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.12/dist-packages (from weasel
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->spacy)
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from smart-open<8.0.0,>=5.2.1-
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
NLTK and spaCy are installed and ready.
```

```
import spacy

# Load the English model
nlp = spacy.load('en_core_web_sm')
```

```
# Process the essay text
doc = nlp(essay_text)

# Extract tokens (words) and filter out punctuation and whitespace
word_tokens = [token.text for token in doc if not token.is_punct and not token.is_space]

print(f"Number of words after tokenization: {len(word_tokens)}")
print("First 20 word tokens:")
print(word_tokens[:20])
```

```
Number of words after tokenization: 261
First 20 word tokens:
['Artificial', 'Intelligence', 'AI', 'is', 'rapidly', 'transforming', 'various', 'sectors', 'and', 'education',
```

```
essay_text = '''Artificial Intelligence (AI) is rapidly transforming various sectors, and education is no exce

Furthermore, AI tools can automate mundane administrative tasks, freeing up educators to focus more on direct

However, the integration of AI in education is not without its challenges. Concerns about data privacy, algori

print("Essay text loaded into 'essay_text' variable.")
```

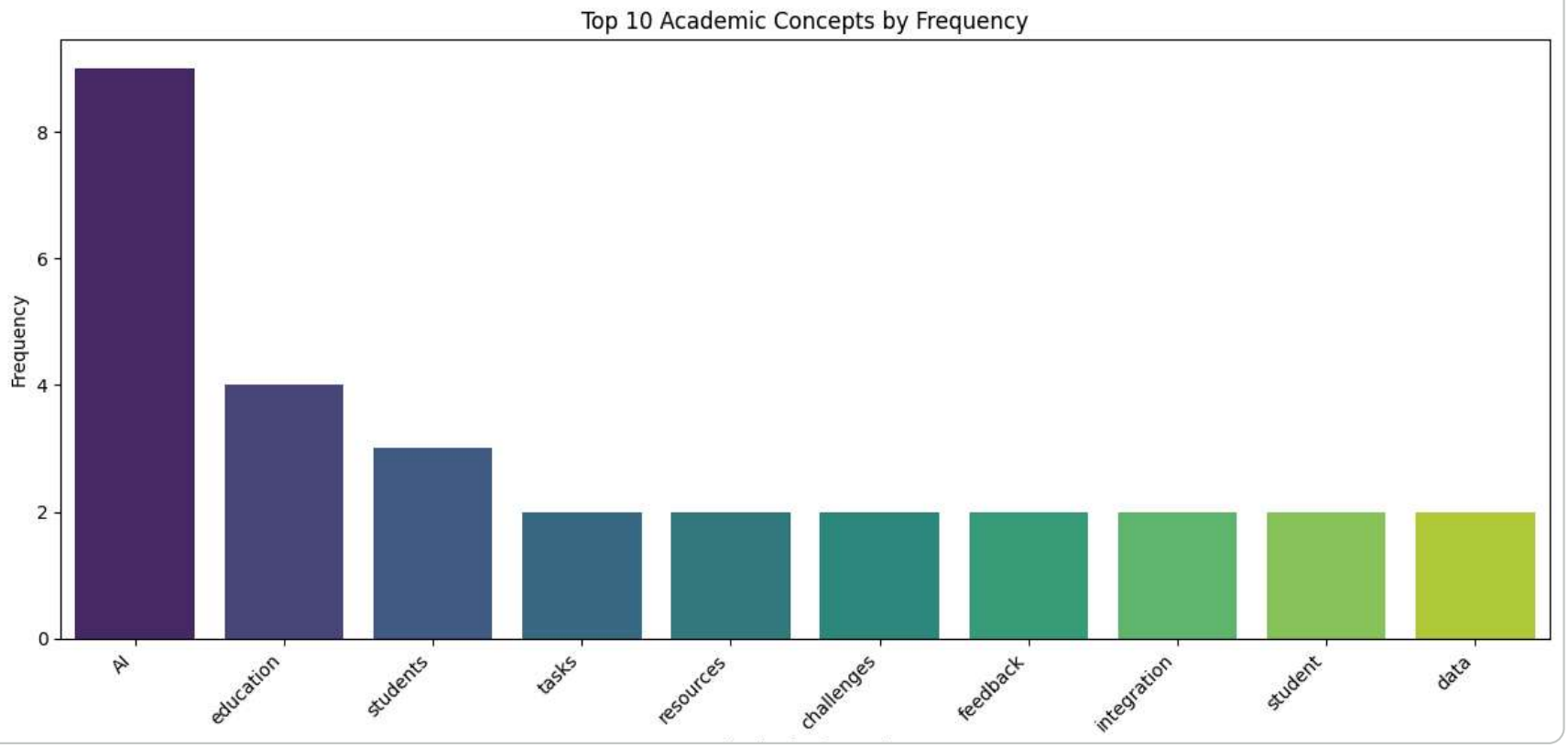
```
Essay text loaded into 'essay_text' variable.
```

```
import nltk

# Download the necessary NLTK resource for POS tagging if not already present
try:
    nltk.data.find('taggers/averaged_perceptron_tagger_eng')
except LookupError:
    print("Downloading 'averaged_perceptron_tagger_eng'...")
    nltk.download('averaged_perceptron_tagger_eng')

# Perform POS tagging using NLTK
pos_tags = nltk.pos_tag(word_tokens)
```

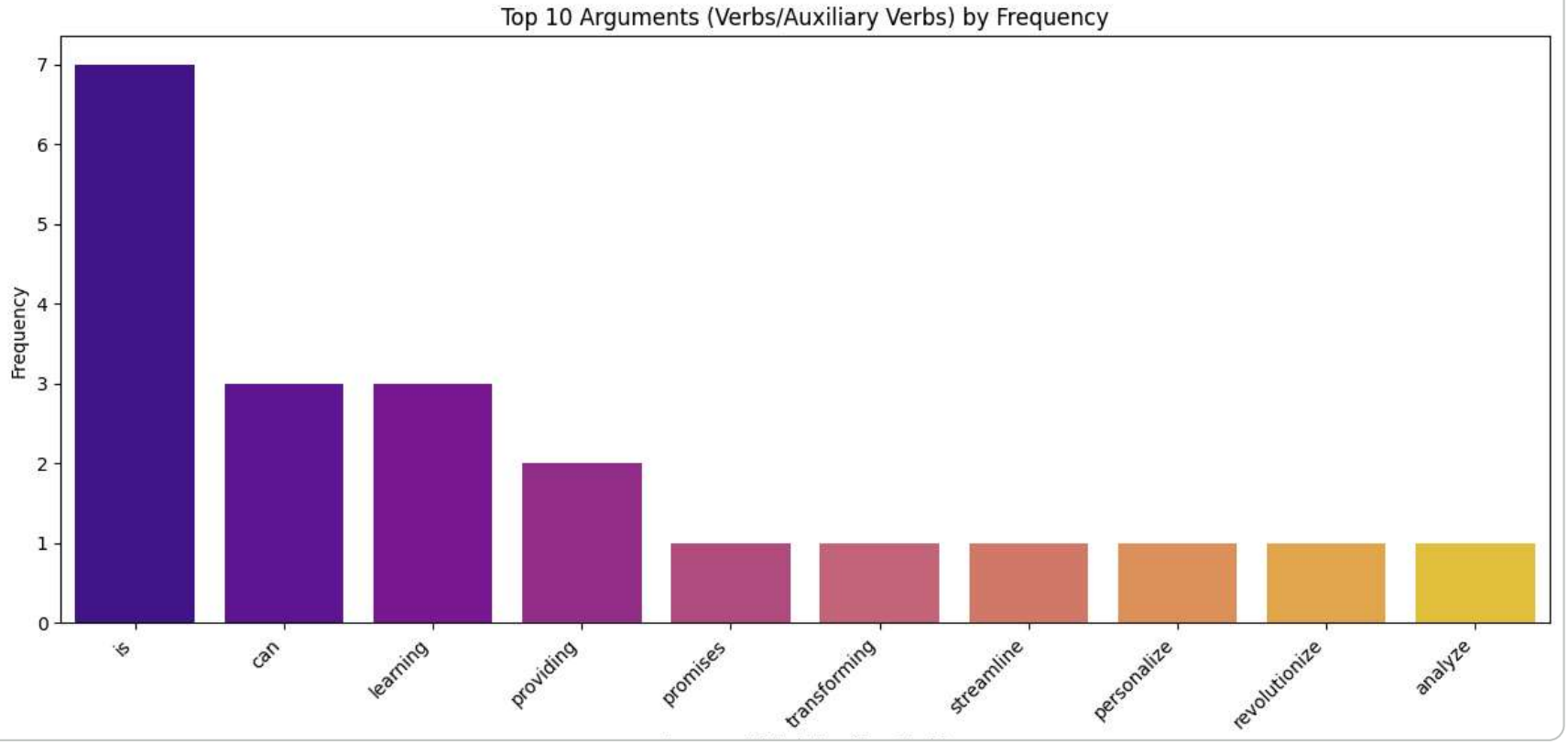
```
print("First 20 POS-tagged tokens:")
print(pos_tags[:20])

# To observe the tag set, we can get unique tags
unique_pos_tags = sorted(list(set([tag for word, tag in pos_tags])))
print(f"\nNumber of unique POS tags observed: {len(unique_pos_tags)}")
print("Unique POS tags observed:")
print(unique_pos_tags)
```

```
Downloading 'averaged_perceptron_tagger_eng'...
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
First 20 POS-tagged tokens:
[('Artificial', 'JJ'), ('Intelligence', 'NNP'), ('AI', 'NNP'), ('is', 'VBZ'), ('rapidly', 'RB'), ('transforming

Number of unique POS tags observed: 23
Unique POS tags observed:
['CC', 'CD', 'DT', 'FW', 'IN', 'JJ', 'MD', 'NN', 'NNP', 'NNS', 'PRP$', 'RB', 'RBR', 'RBS', 'RP', 'TO', 'VB', 'V
```

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

# Visualize top academic concepts
plt.figure(figsize=(12, 6))
sns.barplot(x='Concept', y='Frequency', data=df_concepts.head(10), hue='Concept', palette='viridis', legend=Fa
plt.title('Top 10 Academic Concepts by Frequency')
plt.xlabel('Academic Concept')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```




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

# Visualize top arguments (verbs)
plt.figure(figsize=(12, 6))
sns.barplot(x='Argument', y='Frequency', data=df_arguments.head(10), hue='Argument', palette='plasma', legend=
plt.title('Top 10 Arguments (Verbs/Auxiliary Verbs) by Frequency')
plt.xlabel('Argument (Verb/Auxiliary Verb)')
plt.ylabel('Frequency')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```




```
from collections import Counter
import pandas as pd

# Calculate frequency of academic concepts
concept_frequencies = Counter(academic_concepts)
df_concepts = pd.DataFrame(concept_frequencies.items(), columns=['Concept', 'Frequency'])
df_concepts = df_concepts.sort_values(by='Frequency', ascending=False).reset_index(drop=True)

print("\nTop 20 Academic Concepts by Frequency:")
display(df_concepts.head(20))

# Calculate frequency of arguments
argument_frequencies = Counter(arguments)
df_arguments = pd.DataFrame(argument_frequencies.items(), columns=['Argument', 'Frequency'])
df_arguments = df_arguments.sort_values(by='Frequency', ascending=False).reset_index(drop=True)
```

```
print("\nTop 20 Arguments (Verbs/Auxiliary Verbs) by Frequency:")
display(df_arguments.head(20))
```

```
Top 20 Academic Concepts by Frequency:
```

	Concept	Frequency
0	AI	9
1	education	4
2	students	3
3	tasks	2
4	resources	2
5	challenges	2
6	feedback	2
7	integration	2
8	student	2
9	data	2
10	support	2
11	learning	2
12	sectors	1
13	Intelligence	1
14	Artificial	1
15	experiences	1
16	platforms	1
17	methodologies	1
18	impacts	1

```
# Identify nouns as 'academic concepts' and verbs as 'arguments' using spaCy POS tags
academic_concepts = []
arguments = []
```

```
for word, tag in spacy_pos_tags:
    if tag in ['NOUN', 'PROPN']:
        academic_concepts.append(word)
    elif tag in ['VERB', 'AUX']:
        arguments.append(word)

print("Academic Concepts (Nouns/Proper Nouns):")
print(academic_concepts[:50]) # Displaying first 50 for brevity
print(f"\nTotal academic concepts identified: {len(academic_concepts)}")

print("\nArguments (Verbs/Auxiliary Verbs):")
print(arguments[:50]) # Displaying first 50 for brevity
print(f"\nTotal arguments identified: {len(arguments)}")
```

```
Academic Concepts (Nouns/Proper Nouns):
['Artificial', 'Intelligence', 'AI', 'sectors', 'education', 'exception', 'integration', 'teaching', 'methodolo
Total academic concepts identified: 90

Arguments (Verbs/Auxiliary Verbs):
['is', 'transforming', 'is', 'promises', 'revolutionize', 'personalize', 'streamline', 'is', 'powered', 'can',
Total arguments identified: 49
```

8	revolutionize	1
9	analyze	1
10	identify	1
11	ensures	1
12	receive	1
13	powered	1

```
# Perform Named Entity Recognition (NER) using spaCy
# The 'doc' object already contains the NER information from the previous spaCy processing.

print("Named Entities found in the essay text:")
if doc.ents:
    for ent in doc.ents:
        print(f"  Text: {ent.text:<20} Label: {ent.label_}")
else:
    print("No named entities found.")
```

```
Named Entities found in the essay text:
  Text: Artificial Intelligence (AI Label: ORG
  Text: One                  Label: CARDINAL
  Text: AI                   Label: GPE
```

```
Text: AI                    Label: GPE
Text: AI                    Label: ORG
Text: AI                    Label: GPE
Text: 24/7                  Label: CARDINAL
Text: AI                    Label: GPE
Text: AI                    Label: GPE
Text: AI                    Label: GPE
Text: the 21st century      Label: DATE
```

```
academic_words = set([
    "transforming", "sectors", "integration", "methodologies", "personalization",
    "curricula", "adaptive", "streamline", "administrative", "automate",
    "mundane", "educators", "interaction", "feedback", "accessible",
    "clarification", "challenges", "privacy", "algorithmic", "reliance",
    "equitable", "socioeconomic", "trajectory", "efficient", "reshaping",
    "disseminated"
])

comparison_data = []
# Assuming word_tokens, pos_tags, and spacy_pos_tags are already available from previous steps
for i, word in enumerate(word_tokens):
    if word.lower() in academic_words:
        nltk_tag = pos_tags[i][1] if i < len(pos_tags) else 'N/A'
        spacy_tag = spacy_pos_tags[i][1] if i < len(spacy_pos_tags) else 'N/A'
        comparison_data.append({
            'Word': word,
            'NLTK Tag': nltk_tag,
            'spaCy Universal Tag': spacy_tag
        })

print("POS Tag Comparison for Academic Vocabulary:")
for item in comparison_data:
    print(f"Word: {item['Word']:<20} NLTK: {item['NLTK Tag']:<10} spaCy: {item['spaCy Universal Tag']}")

# Optionally, use pandas for a more structured display
import pandas as pd
if comparison_data:
    df_comparison = pd.DataFrame(comparison_data)
    print("\n--- Detailed Comparison (DataFrame) ---")
```

```
        display(df_comparison)
    else:
        print("No academic words found in the tokenized essay to compare.")
```

```
POS Tag Comparison for Academic Vocabulary:
Word: transforming         NLTK: VBG        spaCy: VERB
Word: sectors              NLTK: NNS        spaCy: NOUN
Word: integration          NLTK: NN         spaCy: NOUN
Word: methodologies        NLTK: NNS        spaCy: NOUN
Word: streamline           NLTK: VB         spaCy: VERB
Word: administrative       NLTK: JJ         spaCy: ADJ
Word: personalization      NLTK: NN         spaCy: NOUN
```