**Implementing Simple NLP Tasks Using Libraries Like NLTK and SpaCy**

NLTK (Natural Language Toolkit) and SpaCy are **popular Python libraries** for performing NLP tasks efficiently. They provide tools for text preprocessing, tokenization, part-of-speech tagging, named entity recognition, and more.

---

**Common NLP Tasks with NLTK and SpaCy**

1. **Tokenization** – Splitting text into words or sentences.
2. **Stop Words Removal** – Removing common words that do not contribute meaningful information.
3. **Stemming and Lemmatization** – Reducing words to their root or dictionary form.
4. **Part-of-Speech (POS) Tagging** – Identifying the grammatical role of each word.
5. **Named Entity Recognition (NER)** – Detecting entities such as names, dates, or organizations.

---

**Python Example Using NLTK**

```
import nltk

from nltk.tokenize import word_tokenize, sent_tokenize

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer, WordNetLemmatizer


# Download required NLTK data

nltk.download('punkt')

nltk.download('stopwords')

nltk.download('wordnet')

nltk.download('averaged_perceptron_tagger')


text = "Apple is looking at buying U.K. startup for $1 billion."
```

```
# Sentence and word tokenization
sentences = sent_tokenize(text)
words = word_tokenize(text)
print("Sentences:", sentences)
print("Words:", words)

# Remove stop words
stop_words = set(stopwords.words('english'))
filtered_words = [w for w in words if w.lower() not in stop_words]
print("Filtered Words:", filtered_words)

# Stemming
stemmer = PorterStemmer()
stemmed_words = [stemmer.stem(w) for w in filtered_words]
print("Stemmed Words:", stemmed_words)

# Lemmatization
lemmatizer = WordNetLemmatizer()
lemmatized_words = [lemmatizer.lemmatize(w) for w in filtered_words]
print("Lemmatized Words:", lemmatized_words)

# Part-of-Speech Tagging
pos_tags = nltk.pos_tag(words)
print("POS Tags:", pos_tags)
```

---

**Python Example Using SpaCy**

```
import spacy
```

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

text = "Apple is looking at buying U.K. startup for $1 billion."
doc = nlp(text)

# Tokenization
tokens = [token.text for token in doc]
print("Tokens:", tokens)

# Lemmatization
lemmas = [token.lemma_ for token in doc]
print("Lemmas:", lemmas)

# Part-of-Speech Tagging
pos_tags = [(token.text, token.pos_) for token in doc]
print("POS Tags:", pos_tags)

# Named Entity Recognition
entities = [(ent.text, ent.label_) for ent in doc.ents]
print("Named Entities:", entities)
```

---

**Key Points**

- NLTK is excellent for **learning and experimentation**; it provides flexibility and extensive resources.
- SpaCy is designed for **high performance and industrial applications**, with faster processing and built-in pipelines.

- Both libraries allow you to **perform fundamental NLP tasks efficiently**, forming the basis for advanced applications like sentiment analysis, chatbots, and text classification.

Implementing these tasks helps in **preprocessing text data**, extracting meaningful information, and preparing it for machine learning or deep learning models.