



Start coding or [generate](#) with AI.

```
import pandas as pd

# Ensure all lists are of the same length for DataFrame creation
min_len = min(len(nltk_word_tokens), len(stemmed_tokens), len(lemmatized_tokens_nltk),

comparison_df = pd.DataFrame({
    'Original Word': nltk_word_tokens[:min_len],
    'Stemmed (Porter)': stemmed_tokens[:min_len],
    'Lemmatized (NLTK)': lemmatized_tokens_nltk[:min_len],
    'Lemmatized (spaCy)': lemmatized_tokens_spacy[:min_len]
})

display(comparison_df.head(20))
```

	Original Word	Stemmed (Porter)	Lemmatized (NLTK)	Lemmatized (spaCy)	
0	Diabetes	diabet	Diabetes	diabetes	
1	is	is	is	be	
2	a	a	a	a	
3	chronic	chronic	chronic	chronic	
4	disease	diseas	disease	disease	
5	that	that	that	that	
6	affects	affect	affect	affect	
7	how	how	how	how	
8	the	the	the	the	
9	body	bodi	body	body	
10	processes	process	process	process	
11	blood	blood	blood	blood	
12	sugar	sugar	sugar	sugar	
13	
14	If	if	If	\n	
15	untreated	untreat	untreated	if	
16	,	,	,	untreat	
17	diabetes	diabet	diabetes	,	
18	may	may	may	diabete	
19	cause	caus	cause	may	

As observed in the comparison table, stemming often reduces words to a form that is not a valid word, such as 'respiratori' from 'respiratory' or 'inflammatori' from 'inflammatory'. While this can be useful for certain text analysis tasks by reducing word variations, it can obscure the original meaning.

In contrast, lemmatization (especially with spaCy) aims to convert words to their base or dictionary form (lemma) while ensuring the result is a valid word. For instance, 'presented' becomes 'present' and 'markers' becomes 'marker'. This preservation of lexical meaning is particularly important in domains like healthcare, where precise terminology is critical. NLTK's lemmatizer, when used without specifying the part-of-speech, might not always produce the desired lemma, as seen with 'presented' remaining unchanged, while spaCy, with its integrated POS tagging, often provides more accurate lemmatization.

```
wordnet_lemmatizer = WordNetLemmatizer()
lemmatized_tokens_nltk = [wordnet_lemmatizer.lemmatize(word) for word in nltk_word_tokens]

print(f"Original NLTK Word Tokens (first 10): {nltk_word_tokens[:10]}...")
print(f"\nLemmatized NLTK Tokens (first 10): {lemmatized_tokens_nltk[:10]}...")
```

```
Original NLTK Word Tokens (first 10): ['Diabetes', 'is', 'a', 'chronic', 'disease', 'that', 'affects', 'the', 'number', 'of']
Lemmatized NLTK Tokens (first 10): ['Diabetes', 'is', 'a', 'chronic', 'disease', 'that', 'affects', 'the', 'number', 'of']
```

```
lemmatized_tokens_spacy = [token.lemma_ for token in spacy_doc]

print(f"Original spaCy Word Tokens (first 10): {spacy_word_tokens[:10]}...")
print(f"\nLemmatized spaCy Tokens (first 10): {lemmatized_tokens_spacy[:10]}...")
```

```
Original spaCy Word Tokens (first 10): ['Diabetes', 'is', 'a', 'chronic', 'disease', 'that', 'affects', 'the', 'number', 'of']
Lemmatized spaCy Tokens (first 10): ['diabetes', 'be', 'a', 'chronic', 'disease', 'that', 'affects', 'the', 'number', 'of']
```

```
porter_stemmer = PorterStemmer()
stemmed_tokens = [porter_stemmer.stem(word) for word in nltk_word_tokens]

print(f"Original NLTK Word Tokens (first 10): {nltk_word_tokens[:10]}...")
print(f"\nStemmed Tokens (first 10): {stemmed_tokens[:10]}...")
```

```
Original NLTK Word Tokens (first 10): ['Diabetes', 'is', 'a', 'chronic', 'disease', 'that', 'affects', 'the', 'number', 'of']
Stemmed Tokens (first 10): ['diabet', 'is', 'a', 'chronic', 'diseas', 'that', 'affect', 'the', 'number', 'of']
```

```
# Tokenize into words using NLTK
nltk_word_tokens = word_tokenize(medical_text)
print(f"NLTK Word Tokens: {nltk_word_tokens[:10]}...")

# Tokenize into sentences using NLTK
```

```
nltk_sent_tokens = nltk.sent_tokenize(medical_text)
print(f"\nNLTK Sentence Tokens: {nltk_sent_tokens}")
```

```
NLTK Word Tokens: ['Diabetes', 'is', 'a', 'chronic', 'disease', 'that', 'affects', 'how
```

```
NLTK Sentence Tokens: ['Diabetes is a chronic disease that affects how the body proces
```

```
# Process the text with spaCy
spacy_doc = nlp(medical_text)

# Tokenize into words using spaCy
spacy_word_tokens = [token.text for token in spacy_doc]
print(f"spaCy Word Tokens: {spacy_word_tokens[:10]}...")

# Tokenize into sentences using spaCy
spacy_sent_tokens = [sent.text for sent in spacy_doc.sents]
print(f"\nspaCy Sentence Tokens: {spacy_sent_tokens}")
```

```
spaCy Word Tokens: ['Diabetes', 'is', 'a', 'chronic', 'disease', 'that', 'affects', 'h
```

```
spaCy Sentence Tokens: ['Diabetes is a chronic disease that affects how the body proce
```

```
medical_text = """Diabetes is a chronic disease that affects how the body processes b
If untreated, diabetes may cause heart disease, kidney failure, nerve damage and visio
Early diagnosis and proper treatment help improve patient outcomes.
"""
```

The medical text has been loaded into the `medical_text` variable. You can display it to verify the content:

```
print(medical_text)
```

```
Diabetes is a chronic disease that affects how the body processes blood sugar.
If untreated, diabetes may cause heart disease, kidney failure, nerve damage and visio
Early diagnosis and proper treatment help improve patient outcomes.
```

```
!pip install nltk spacy
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.12/dist-packages (3.9.1)
Requirement already satisfied: spacy in /usr/local/lib/python3.12/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages (from
Requirement already satisfied: joblib in /usr/local/lib/python3.12/dist-packages (from
Requirement already satisfied: regex<=2021.8.3 in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (from n
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.12
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.12
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.12/c
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.12/dist
Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/python3.12/dist-pa
```

```
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.12/dist-
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.12/di
Requirement already satisfied: weasel<0.5.0,>=0.4.2 in /usr/local/lib/python3.12/dist-
Requirement already satisfied: typer-slim<1.0.0,>=0.3.0 in /usr/local/lib/python3.12/d
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.12/dist-package
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.12/di
Requirement already satisfied: pydantic!=1.8,!1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (-
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.12/dis-
Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/python3.12/dist
Requirement already satisfied: typing-extensions>=4.14.1 in /usr/local/lib/python3.12/c
Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib/python3.12/d
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.12/d
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.12/d
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.12,
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.12/d
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from
```

```
!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.tar.gz 12.8/12.8 MB 104.0 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 nltk
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('punkt_tab') # Added to resolve the LookupError
```

```
import spacy
nlp = spacy.load('en_core_web_sm')
```

```
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data] Unzipping tokenizers/punkt_tab.zip.
```

