Text Preprocessing:

Why Text Preprocessing?

In NLP, raw text data is **unstructured and noisy**, which makes it difficult for models to learn meaningful patterns. Preprocessing helps transform this raw data into a **clean, structured, and useful format** for training language models (including LLMs used in Gen AI).

Common Text Preprocessing Steps:

Lowercasing

- Convert all text to lowercase to ensure uniformity.

```
text = "Hello World!"

text = text.lower()

# Output: 'hello world!'
```

Removing Punctuation

- Removes symbols like .,!?@#$ that may not carry meaning (depends on the task).

```
import string

text = "Hello, World!"

text = text.translate(str.maketrans('', '', string.punctuation))

# Output: 'Hello World'
```

Tokenization

- Split text into **tokens** (words, subwords, or characters).

```
from nltk.tokenize import word_tokenize

text = "NLP is awesome!"

tokens = word_tokenize(text)

# Output: ['NLP', 'is', 'awesome', '!']
```

Removing Stopwords

- These are common words like *"the"*, *"is"*, *"and"* that often don't add meaning.

```
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))

filtered_tokens = [w for w in tokens if w.lower() not in stop_words]

# Output: ['NLP', 'awesome', '!']
```

Stemming

- Reduce words to their base form (e.g., "running" → "run").

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()

stemmed = [stemmer.stem(word) for word in filtered_tokens]

# Output: ['nlp', 'awesom', '!']
```

Lemmatization

- Like stemming but uses vocabulary and grammar to find actual root words.

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

lemmatized = [lemmatizer.lemmatize(word) for word in filtered_tokens]

# Output: ['NLP', 'awesome', '!']
```

Removing Numbers

- In some cases, numeric data may not be useful (e.g., years, quantities).

```
import re

text = re.sub(r'\d+', '', text)
```

Removing Extra Whitespaces

```
text = "   NLP     with Gen AI    "

text = ' '.join(text.split())

# Output: "NLP with Gen AI"
```

When Working with Gen AI (LLMs)

If you're using a Generative AI model (like GPT or BERT-based models), you usually don't need extensive preprocessing because:

- These models are trained on noisy, raw text.
- They use tokenizers (e.g., BPE, WordPiece) internally.
- Too much preprocessing may remove useful context.

Tip: For fine-tuning or training from scratch, preprocessing becomes more important. But for inference or prompt engineering, minimal cleaning is usually best.

Summary Table

Step	**Purpose**	**Used In Gen AI?**
Lowercasing	Uniform text format	Sometimes
Removing punctuation	Reduce noise	Sometimes
Tokenization	Break text into analyzable units	Always (internally)
Stopword removal	Focus on meaningful words	Not always
Stemming	Reduce to root forms	Rarely
Lemmatization	Get valid base forms	Rarely
Remove numbers	Remove irrelevant numeric info	Sometimes
Whitespace cleanup	Clean display and structure	Often