🔹 Q.1 Define NLP and Explain the Steps in Text Processing with a Suitable Example

---

✅ What is Natural Language Processing (NLP)?

Natural Language Processing (NLP) is a subfield of artificial intelligence (AI) and computational linguistics that focuses on enabling computers to **understand, interpret, and generate human language** in a meaningful way.

NLP bridges **human communication** (language) and **machine understanding** (data structures, algorithms).

✅ Applications of NLP:

- Chatbots and virtual assistants (e.g., Siri, Alexa, ChatGPT)
- Machine translation (Google Translate)
- Sentiment analysis
- Spam filtering
- Speech recognition

📌 Key Steps in Text Processing (Pipeline):

---

1. Text Acquisition / Input

- Collect raw data (text files, tweets, documents, etc.)

Example:
Input Sentence – `"NLP is transforming the world!"`

---

2. Text Cleaning / Preprocessing

- Remove noise like punctuation, special characters, HTML tags, etc.

Cleaned Text – `"nlp is transforming the world"`

---

3. Tokenization

- Splitting text into smaller units like words or sentences.

Tokenized – `[‘nlp’, ‘is’, ‘transforming’, ‘the’, ‘world’]`

---

4. Normalization

- Convert to lowercase, remove stopwords, expand contractions, etc.

Normalized – `[‘nlp’, ‘transforming’, ‘world’]`

---

5. Stemming / Lemmatization

- Reduce words to their root forms.

Example:

- Stemming: `transforming` → `transform`
- Lemmatization: `better` → `good`

---

6. POS Tagging (Part-of-Speech)

- Assign grammatical tags to each word.

Example:

- `nlp/NN`, `is/VBZ`, `transforming/VBG`, `world/NN`

---

7. Named Entity Recognition (NER)

- Identify entities like names, places, organizations.

Example: `"Apple is based in California."`
Entities: `Apple` → ORG, `California` → LOC

---

8. Vectorization / Feature Extraction

- Convert text into numerical features using:
  - Bag of Words (BoW)
  - TF-IDF
  - Word embeddings (Word2Vec, BERT)

✅ Summary Flow:

```
Raw Text → Cleaning → Tokenization → Normalization → POS/NER → Vectorization → Model Inpu
```

📌 Example Recap:

Text: `"NLP is transforming the world!"`

After processing:

- Tokens: `['nlp', 'transform']`
- Vector: `[0.45, 0.22, 0.13, …]` → can be fed into ML/DL models

🔹 Q.2 Define Empirical Laws (e.g., Zipf's Law, Heaps' Law) in NLP. State Their Significance in Corpus Analysis

---

✅ What Are Empirical Laws in NLP?

Empirical laws describe **observed statistical patterns** in natural language corpora. These laws are useful for understanding and optimizing how language data behaves in real-world NLP applications.

📘 1. Zipf's Law

> "In any large natural language corpus, the frequency of a word is **inversely proportional** to its rank."

Formula:

$$f(r) \propto \frac{1}{r^s}$$

Where:

- $f(r)$: frequency of the word at rank r
- $s \approx 1$ for natural language

🔍 Example:

Rank	Word	Frequency
1	the	5000
2	of	2500
3	and	1700

> The second most frequent word appears ~**half as often** as the first, and so on.

✅ Significance:

- Helps in **vocabulary compression**.
- Basis for **language model optimizations**.

- Explains the **long-tail distribution** of rare words.

📘 2. Heaps' Law

> "As more text is processed, the **number of unique words (vocabulary)** grows, but at a **sub-linear rate**."

Formula:

$$V(n) = K \cdot n^{\beta}$$

Where:

- $V(n)$: vocabulary size
- n: total number of words in the corpus
- K, β: constants (typically β ≈ 0.4–0.6)

🔍 Example:

Tokens (n)	Unique Words (V)
10,000	2,000
100,000	6,000
1,000,000	20,000

> Vocabulary grows, but **not proportionally** to the total size.

✅ Significance:

- Important for estimating **storage requirements**.
- Helps in **vocabulary pruning** for model design.
- Shows that even **large corpora have manageable vocabulary size**.

🧠 Together, These Laws Help NLP Engineers:

- Design **efficient language models**
- Optimize **indexing and storage**
- Choose **cutoffs for rare words**
- Understand the **sparsity** of natural language data