1. What is NLP?

- **Definition**: Natural Language Processing (NLP) is the field at the intersection of computer science and linguistics, enabling machines to understand, interpret, and generate human language.
- **Goal**: It involves building models that map between raw text and structured representations like tags, syntax, meanings, or embeddings.
- **Levels**:
    - *Syntax*: tokenizing, parsing, part-of-speech tagging
    - *Semantics*: understanding meaning—named entities, sentiment
    - *Pragmatics / Discourse*: context-aware dialogue, coreference

2. Why NLP Matters

- **Ubiquity of text**: With massive volumes of text data (articles, chats, social media), NLP unlocks insights and automation.
- **Real-world utility**:
    - Sentiment analysis: gauging customer opinions
    - Chatbots & virtual assistants
    - Machine translation (e.g., English→Spanish)
    - Speech recognition and text-to-speech

3. Core Applications Covered

- **Text classification**: Spam detection, topics, mood, intent.
- **Named Entity Recognition (NER)**: Extracting names of people, places, brands.
- **Part-of-Speech tagging & parsing**: Grammatical structure, useful for translation and grammar checks.
- **Vector embedding models**: Word2Vec, fastText, transformer embeddings for similarity and understanding.
- **Sequence modeling**: RNNs, LSTMs, Transformers for generation and translation.
- **Advanced projects**: Building RAG (Retrieval-Augmented Generation) chatbots by indexing documents and conditioning language models.

4. NLP Workflow / Pipeline

The videos frequently return to this typical structure:

1. **Data gathering**
    - Collect raw text (web scraping, corpora).
    - Clean & preprocess: lowercasing, removing punctuation, normalizing.

2. **Tokenization**
    - Breaking text into words, subwords, or characters.

3. **Text representation (feature extraction)**
    - Count vectors (bag-of-words), TF-IDF.
    - Pre-trained embeddings (Word2Vec, GPT/BERT tokens).
    - Domain-specific (e.g., Indian languages via iNLTK).

4. **Modeling**
    - Traditional: Naive Bayes, SVMs, logistic regression.
    - Neural methods: RNNs → LSTMs/GRUs → Transformers.
    - Task-specific heads: classification, sequence-tagging, generation.

5. **Training & evaluation**
    - Split data (train/val/test), tune hyperparameters.
    - Metrics: accuracy, F1 for classification, BLEU/ROUGE for generation tasks.

6. **Deployment**
    - Export model (pickle, ONNX).
    - Wrap in API (FastAPI, Flask) or chatbot interface.
    - Monitor performance, retrain with fresh data.

5. Step-by-Step: From Data to App

- **Data acquisition & preprocessing**: Web scrape, clean, tokenize.
- **Feature engineering**: Choose representation—TF-IDF or embeddings.
- **Model choice**:
    - Intro models: Naive Bayes, logistic regression.
    - Advanced: sliding window neural nets, RNNs.
    - Cutting edge: Transformers (BERT, GPT-style).

- **Fine-tuning & evaluation**.
- **Application layer**: Build chatbots or other interactives.

6. Special Topics

- **Indic-language support (iNLTK)**:
    - Provides pre-trained tokenizers and embeddings in 13 Indic languages.
    - Enables strong classification performance with limited data.
- **Building RAG-powered chatbots**:
    - Index relevant docs, retrieve on query.
    - Feed retrieved context to LLM for grounded responses.