**NLP Basics**

**Text Preprocessing**

* Removing Punctuation and Stop Words: Eliminating punctuation marks and common words (stop words) to focus on meaningful content.

Input: "NLP is fascinating! But it's also challenging."

Output: "NLP fascinating also challenging"

* Tokenization: Dividing text into individual words or tokens for analysis.

Input: "Text analysis is important."

Output: ["Text", "analysis", "is", "important"]

* Stemming and Lemmatization: Reducing words to their base form (stemming) or dictionary form (lemmatization) for consistency.

Input (Stemming): "Jumps, jumping, jumped"

Output (Stemming): "jump, jump, jump"

Input (Lemmatization): "Jumps, jumping, jumped"

Output (Lemmatization): "jump, jump, jump"

* Text Normalization: Ensuring uniform text formats, like converting uppercase to lowercase.

Input: "UPPERCASE and lowercase words"

Output: "uppercase and lowercase words"

**Advanced Text Preprocessing**

* POS Tagging (Part of Speech Tagging): Assigning grammatical categories (e.g., noun, verb) to words in a sentence.

Input: "I am reading a book."

Output: [("I", "PRON"), ("am", "VERB"), ("reading", "VERB"), ("a", "DET"), ("book", "NOUN")]

* Parsing: Analyzing sentence structure to understand relationships between words.

Input: "The cat chased the mouse."

Output: Parsing tree showing the grammatical structure of the sentence.

(S (NP (DT The) (NN cat)) (VP (VBD chased) (NP (DT the) (NN mouse))))

* Coreference Resolution: Identifying and linking words or phrases that refer to the same entity in a text.

Input: "John said he is coming. He is bringing his guitar."

Output: Coreference resolution links "John" to "he" and "he" to "his guitar."

**Text Analytics**

**Text Representation**

* One-Hot Encoding: Converting words into binary vectors where each word is represented by a unique position.

Input: "I love NLP."

Output: [1, 0, 0, 0, 0] (1 for "I," 0s for other words in the vocabulary)

* Bag of Words: Representing text as a collection of words, disregarding order, and using word frequencies.

Input: "Text analysis is important. Text is fascinating."

Output:

{"Text": 2, "analysis": 1, "is": 2, "important": 1, "fascinating": 1}

* Tf-idf (Term Frequency-Inverse Document Frequency): Assigning weights to words based on their importance in a document relative to a corpus.

Input: A collection of documents

Output: Weighted representation of words based on their importance in documents relative to the entire corpus.

Example for the term "NLP" in document 1: 0.1

Example for the term "NLP" in document 2: 0.05

* N-grams (Unigram and Bigram): Capturing sequences of words (unigram, bigram, etc.) to understand context.

Input: "Natural language processing is fun."

Output (Unigrams): ["Natural", "language", "processing", "is", "fun"]

Output (Bigrams): ["Natural language", "language processing", "processing is", "is fun"]

* Word Embedding: Representing words as continuous vectors in a high-dimensional space, preserving semantic relationships.

Input: Word2Vec model trained on a large text corpus

Output: Word vectors, e.g., "king" - "man" + "woman" = "queen"

**Libraries**

* NLTK (Natural Language Toolkit): A Python library for NLP that offers tools for text preprocessing, linguistic data, and various NLP tasks.
* Spacy: A popular NLP library that provides efficient and accurate linguistic analysis, including tokenization, POS tagging, and named entity recognition.
* Gensim: A library for topic modeling and word embedding, particularly useful for creating word vectors and working with large text corpora.