1. What are various text similarity measures? Explain any two of them.

Text Similarity Measures
Text similarity measures are mathematical techniques used to determine how "close" or similar two pieces of text (documents, sentences, or phrases) are in terms of content or meaning. These are crucial in applications like search engines, plagiarism detection, and recommendation systems.
Common Measures:

1. **Cosine Similarity** (Vector-based)
2. **Jaccard Similarity** (Set-based)
3. **Euclidean Distance** (Distance-based)
4. **Manhattan Distance** (Distance-based)
5. **Levenshtein Distance** (Edit-based, for strings)

Explanation of Two Measures:

1. Cosine Similarity
Cosine similarity measures the cosine of the angle between two non-zero vectors in a multi-dimensional space. In text processing, documents are represented as vectors of word frequencies.

- **Concept:** It focuses on the **orientation** (direction) of the vectors rather than their magnitude. This is important because two documents might talk about the same topic (have similar word distribution) but differ vastly in length.
- Formula:

$$\text{Cosine Similarity}(A, B) = \frac{A \cdot B}{||A|| \times ||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

- **Interpretation:**
  - **1:** Vectors point in the exact same direction (Identical content).
  - **0:** Vectors are orthogonal (90 degrees) (No shared words).
  - **-1:** Opposite directions (Not usually possible in standard text counts).

2. Jaccard Similarity
Jaccard similarity measures the similarity between two sets of data. It treats the text documents as "bags of words" (sets) without worrying about the frequency of the words.
- **Concept:** It looks at the overlap of unique words between the two documents relative to the total number of unique words across both documents.
- Formula:

- **Formula:**

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\text{Intersection}}{\text{Union}}$$

- **Example:**
  - Doc A: "AI is great" $\rightarrow$ {AI, is, great}
  - Doc B: "AI is bad" $\rightarrow$ {AI, is, bad}
  - Intersection: {AI, is} (Size 2)
  - Union: {AI, is, great, bad} (Size 4)
  - Jaccard Score: $2/4 = 0.5$.

---

2. Write short notes on: i) Stemming ii) Lemmatization iii) Topic modeling

i) Stemming

- **Definition:** Stemming is a rudimentary text normalization process that cuts off the ends of words (suffixes) to reduce them to their base form or "root."
- **Mechanism:** It uses heuristic rules (simple algorithms) to chop off characters. It does not check a dictionary.
- **Pros:** Very fast and computationally inexpensive.
- **Cons:** It can produce non-words. For example, "Universities" might become "Univers". "Better" might become "Bet".
- **Algorithm:** The most famous algorithm is the **Porter Stemmer**.
- **Example:**
  - Input: "Running, Runs, Runner"
  - Output: "Run, Run, Run"

ii) Lemmatization

- **Definition:** Lemmatization is a sophisticated linguistic process that groups together the inflected forms of a word so they can be analyzed as a single item, identified by the word's lemma (dictionary form).
- **Mechanism:** It performs a full morphological analysis of the word. It looks up the word in a dictionary (like WordNet) and considers the context (Part of Speech).
- **Pros:** Highly accurate; always returns a valid word.
- **Cons:** Slower than stemming because it requires dictionary lookups.

- **Example:**
  - Input: "Better"
  - Output: "Good" (Stemming would fail here).
  - Input: "Running" (Verb) —> "Run".

iii) Topic Modeling

- **Definition:** Topic modeling is an unsupervised machine learning technique used to discover abstract "topics" that occur in a collection of documents.
- **Concept:** It assumes that every document is a mixture of topics and every topic is a mixture of words. It tries to reverse-engineer this structure to find hidden patterns.
- **Algorithm:** The most common algorithm is **Latent Dirichlet Allocation (LDA)**.
- **Use Case:** If you have 10,000 news articles, Topic Modeling can automatically group them into "Sports," "Politics," and "Finance" without you ever reading them.

---

3. Explain the process of text processing.

Text Processing Pipeline
The process of text processing (or Natural Language Processing pipeline) transforms unstructured text data into a structured format that machines can understand and analyze.
Steps in the Process:

1. **Data Acquisition:** Collecting raw text from sources (web scraping, database dumps, PDF files, APIs).
2. **Text Cleaning:**
   - Removing HTML tags (if web data).
   - Removing special characters, emojis, and punctuation.
   - Removing numbers (unless relevant).
3. **Normalization:**
   - **Lowercasing:** Converting "Hello" and "hello" to the same form to avoid duplication.
   - **Spell Checking:** Correcting typos.
4. **Tokenization:** Splitting the text into smaller units called tokens (words or sentences).
5. **Stop Word Removal:**
   - Removing common words that carry little semantic meaning (e.g., "the", "is", "at", "on"). This reduces the dataset size significantly.
6. **Stemming/Lemmatization:** Reducing words to their root forms (e.g., "flying" —> "fly") to consolidate features.
7. **Feature Extraction (Vectorization):**
   - Converting the cleaned text into numerical vectors.
   - Methods: Bag of Words (BoW), TF-IDF, Word Embeddings (Word2Vec, GloVe).

---

4. Write short notes on: i) Features selection ii) Document

representation

i) Feature Selection in NLP

- **Definition:** Feature selection involves selecting a subset of relevant features (words/n-grams) for use in model construction. In NLP, the number of features (vocabulary size) can be massive (100,000+ words).
- **Goal:** To reduce the dimensionality of the dataset, remove noise (irrelevant words), and speed up training time without sacrificing accuracy.
- **Methods:**
  - **Frequency-based:** Removing words that appear too rarely (typos) or too frequently (stopwords).
  - **Chi-Square Test:** Measures the independence between a feature (word) and the class (category).
  - **Information Gain:** Measures how much information a feature provides about the class.

ii) Document Representation

- **Definition:** Since computers cannot understand strings of text, documents must be converted into numerical formats (vectors).
- **Approaches:**
  1. **One-Hot Encoding:** A massive vector of 0s and a single 1. Very sparse and inefficient.
  2. **Bag of Words (BoW):** Counts the frequency of words in a document. Ignores grammar and order.
  3. **TF-IDF (Term Frequency-Inverse Document Frequency):** Weighs words based on how unique they are to a specific document.
  4. **Word Embeddings:** Dense vector representations (like Word2Vec or BERT) where words with similar meanings have similar vector values.

---

5. Describe tokenization with an example.

Tokenization
Tokenization is the very first step in NLP. It is the process of breaking a stream of text into smaller chunks, known as tokens. These tokens can be words, characters, or sub-words.
Types of Tokenization:

1. **White Space Tokenization:** Splits text whenever it sees a space.
2. **Sentence Tokenization:** Splits text into distinct sentences.
3. **Word Tokenization:** Separates words and punctuation marks properly.

Detailed Example:
Consider the string: "Dr. Smith said, 'I'm coming home!'"

- **Naive Split (Space):** ["Dr.", "Smith", "said,", "'I'm", "coming", "home!'"]

- ○ *Issue:* Punctuation is attached to words ("said,").
- **Proper Word Tokenization:** ["Dr.", "Smith", "said", ",", """, "I", "'m", "coming", "home", "!", """]
  - *Note:* The contraction "I'm" is split into "I" and "'m" (am), and punctuation is treated as separate tokens.

---

6. What is preprocessing? Explain method of text preprocessing.

Preprocessing
Preprocessing is the series of steps taken to clean and prepare raw text data for analysis. Real-world data is often incomplete, inconsistent, and lacking in specific behaviors or trends (noisy).
Methods of Text Preprocessing (Detailed Breakdown):

1. **Lowercasing:**
   - Converting all text to lower case.
   - *Reason:* The computer treats "Apple" and "apple" as two different words. Lowercasing unifies them.
2. **Noise Removal:**
   - Removing headers, footers, HTML tags, and metadata (especially in web scraping).
   - Using Regular Expressions (Regex) to strip out special characters or digits.
3. **Tokenization:** (As explained in Q5).
4. **Stop Word Removal:**
   - Filtering out high-frequency words (English has ~400 common stop words like "the", "and", "in").
   - *Reason:* These words dominate the frequency count but contribute little to the topic/sentiment.
5. **Stemming/Lemmatization:** (As explained in Q2).
6. **N-gram Generation:**
   - Grouping words together.
   - Unigram: "New", "York". Bigram: "New York".
   - *Reason:* Captures context that single words might miss (e.g., "Not good").

---

7. Explain topic modeling with Latent Dirichlet Allocation algorithm.

Latent Dirichlet Allocation (LDA)
LDA is a popular generative statistical model used for Topic Modeling. It allows sets of observations (documents) to be explained by unobserved groups (topics).
Core Assumptions of LDA:

1. **Documents are mixtures of topics:** A single news article isn't just about "Sports." It might be 70% "Sports," 20% "Health" (injuries), and 10% "Finance" (player contracts).
2. **Topics are mixtures of words:** The topic "Sports" has a high probability of containing

words like "ball," "score," and "game." The topic "Finance" has words like "dollar," "stock," and "market."

The Process (Simplified):

1. **Input:** A collection of documents (Corpus) and a specified number of topics, K.
2. **Initialization:** Randomly assign each word in each document to one of the K topics.
3. **Iterative Learning (Gibbs Sampling):**
   - The algorithm iterates through every word in every document.
   - For a specific word w, it calculates:
     - P(Topic | Document): How prevalent is this topic in this document?
     - P(Word | Topic): How prevalent is this word in this topic?
   - It reassigns the word w to a new topic based on these probabilities.
4. **Convergence:** After many iterations, the topic assignments stabilize.

Output:

- A list of Topics, where each topic is a list of words with associated probabilities.

---

8. Write note on feature extraction. What are the practical uses of feature extraction?

Feature Extraction
Feature extraction (or Vectorization) in NLP is the process of transforming processed text data into numerical feature vectors that machine learning algorithms can understand. Algorithms cannot do math on strings like "hello"; they need numbers like [0.1, 0.5, 0.0].
Techniques:

1. **Bag of Words (BoW):** Creates a matrix where rows are documents and columns are words. The values are simple word counts.
2. **TF-IDF:** Transforms counts into weighted scores, penalizing words that appear in *every* document (like "the") and boosting rare, distinct words.
3. **Word Embeddings:** Maps words to vectors of real numbers in a continuous space (e.g., Word2Vec), capturing semantic meaning.

Practical Uses of Feature Extraction:

1. **Spam Detection:** Converting email text into vectors to classify them as "Spam" or "Ham" based on the presence of words like "Free," "Winner," or "Click here."
2. **Sentiment Analysis:** Extracting features from customer reviews to determine if the sentiment is Positive, Negative, or Neutral.
3. **Document Clustering:** Grouping similar documents together (e.g., grouping legal documents by case type) based on vector similarity.
4. **Recommender Systems:** Extracting features from product descriptions to recommend similar items to users (e.g., Netflix suggesting movies based on plot summaries).

5. **Language Translation:** Deep learning models use feature extraction (embeddings) to map words from one language to another.