## 1. Jaccard Distance

Jaccard distance is used to measure the dissimilarity between two sets. It's based on the Jaccard index, which calculates similarity between finite sample sets.

The formula for Jaccard distance between two sets $A$ and $B$ is:

$$d_{\text{Jaccard}}(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

Where:

- $|A \cap B|$ is the size of the intersection of sets $A$ and $B$,
- $|A \cup B|$ is the size of the union of sets $A$ and $B$.

**Example**

Let's say:

- $A = \{1, 2, 3\}$
- $B = \{2, 3, 4, 5\}$

Then:

- $|A \cap B| = \{2, 3\}$ (size = 2)
- $|A \cup B| = \{1, 2, 3, 4, 5\}$ (size = 5)

The Jaccard distance is:

$$d_{\text{Jaccard}}(A, B) = 1 - \frac{2}{5} = 1 - 0.4 = 0.6$$

**Characteristics**

- Suitable for binary or categorical data, especially when data can be represented as sets (like user preferences).
- Often used in text analysis, document sim ↓ ty, and recommendation systems.

- Often used in text analysis, document similarity, and recommendation systems.

---

## 2. Cosine Distance

Cosine distance measures the cosine of the angle between two non-zero vectors. It's often used to find similarity between documents or text data.

The formula for cosine similarity between two vectors $A$ and $B$ is:

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\|\|B\|}$$

Then, the **cosine distance** is calculated as:

$$d_{\text{Cosine}}(A, B) = 1 - \text{cosine\_similarity}(A, B)$$

Where:

- $A \cdot B$ is the dot product of vectors $A$ and $B$,
- $\|A\|$ and $\|B\|$ are the magnitudes of $A$ and $B$.

**Example**

If $A = [1, 0, 1]$ and $B = [1, 1, 0]$:

1. Compute the dot product: $A \cdot B = (1 \cdot 1) + (0 \cdot 1) + (1 \cdot 0) = 1$.
2. Find magnitudes: $\|A\| = \sqrt{1^2 + 0^2 + 1^2} = \sqrt{2}$ and $\|B\| = \sqrt{1^2 + 1^2 + 0^2} = \sqrt{2}$.
3. Cosine similarity: $\frac{1}{\sqrt{2} \cdot \sqrt{2}} = 0.5$.
4. Cosine distance: $d_{\text{Cosine}}(A, B) = 1 - 0.5 = 0.5$.

**Characteristics**

- Typically used for text and high-dimensional data (like TF-IDF vectors in NLP).
- Focuses on the direction rather than the magnitude, making it suitable for measuring similarity in sparse data.

### 4. Hamming Distance

Hamming distance calculates the number of positions at which the corresponding elements of two strings of equal length are different. It's typically used for binary data and error detection.

The formula for Hamming distance between two strings $A$ and $B$ of equal length is:

$$d_{\text{Hamming}}(A, B) = \sum_{i=1}^{n} (A_i \neq B_i)$$

Where $n$ is the length of the strings, and $(A_i \neq B_i)$ is 1 if the characters at position $i$ are different, and 0 if they are the same.

**Example**

Consider:

- $A = \text{"}1011101\text{"}$
- $B = \text{"}1001001\text{"}$

The Hamming distance is calculated by comparing each position:

1. Differences at positions 2, 4, and 5.

Thus, the Hamming distance is **3**.

**Characteristics**

- Often used in error detection and correction, where binary or fixed-length strings are analyzed.
- Not applicable if the strings/vectors are of unequal length.

### 3. Edit Distance (Levenshtein Distance)

Edit distance measures how dissimilar two strings are by counting the minimum number of operations (insertions, deletions, substitutions) required to transform one string into the other. The Levenshtein distance is the most common type of edit distance.

**Example**

Consider the strings:

- $\text{"kitten"}$
- $\text{"sitting"}$

The minimum operations are:

1. Substitute "k" with "s": "kitten" → "sitten"
2. Substitute "e" with "i": "sitten" → "sittin"
3. Insert "g" at the end: "sittin" → "sitting"

Thus, the edit distance is **3**.

**Characteristics**

- Commonly used in spell checkers, DNA sequencing, and natural language processing.
- Suitable for scenarios where data is in the form of sequences (like strings).

---