**📄 Script Documentation: Comparing Text Similarity in Python**

**📦 1. Installing Required Libraries**

!pip install scikit-learn sentence-transformers

This command installs the following packages:

* **scikit-learn**: used for machine learning utilities such as TF-IDF vectorization and cosine similarity.
* **sentence-transformers**: enables the use of transformer-based models like BERT for extracting sentence-level embeddings.

**📚 2. Importing Libraries**

from sklearn.feature\_extraction.text import CountVectorizer, TfidfVectorizer

from sklearn.metrics.pairwise import cosine\_similarity

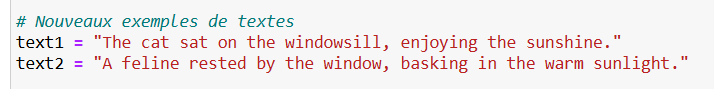
from sentence\_transformers import SentenceTransformer

import numpy as np

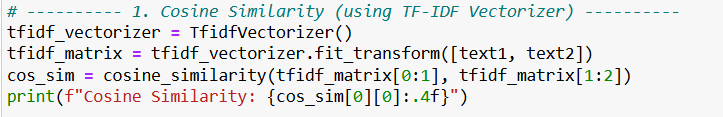
These imports allow:

* **Vectorization** of text (CountVectorizer, TfidfVectorizer)
* **Pairwise similarity computation** (cosine\_similarity)
* **Semantic encoding** using Transformer models (SentenceTransformer)

**📝 3. Input Text Samples**

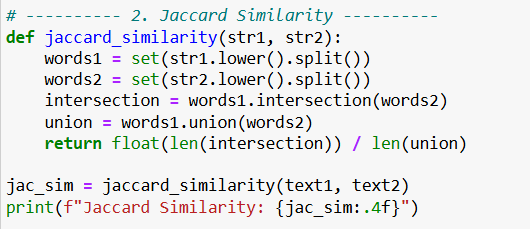


Two similar sentences with different vocabulary are defined. The goal is to evaluate how similar they are using various methods.

**🔍 4. Method 1 – Cosine Similarity with TF-IDF**

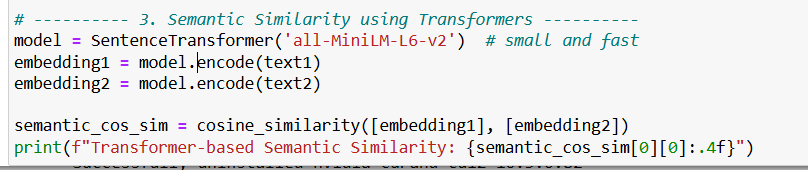
* **TF-IDF** (Term Frequency-Inverse Document Frequency) weights words based on their importance in the corpus.
* **Cosine similarity** computes the angle between two vectorized texts.
* **Output**: A score between 0 and 1 (closer to 1 means more similar).

**🔣 5. Method 2 – Jaccard Similarity**



* **Jaccard Similarity** measures the ratio between the intersection and union of word sets.
* Simple method that does not account for meaning, only exact word overlaps.
* Best for basic lexical comparisons.

**🤖 6. Method 3 – Semantic Similarity using Sentence Transformers**



Uses a **pretrained transformer model**: all-MiniLM-L6-v2 (lightweight and efficient).

Each sentence is converted into a high-dimensional vector (embedding).

**Cosine similarity** is applied to those semantic embeddings.

Captures the **meaning** of sentences rather than just word overlap.

**✅ Sample Output**

Cosine Similarity: 0.3667

Jaccard Similarity: 0.1154

Transformer-based Semantic Similarity: 0.6684

**TF-IDF Cosine**: Moderate similarity based on shared vocabulary.

**Jaccard**: Low similarity due to few overlapping exact words.

**Transformer-based**: High similarity reflecting semantic closeness.

| **Method** | **Type of Similarity** | **Captures Meaning?** | **Description** |
| --- | --- | --- | --- |
| TF-IDF + Cosine Similarity | Statistical | ❌ No | Based on word importance |
| Jaccard Similarity | Lexical | ❌ No | Exact word overlap |
| Transformers + Cosine | Semantic | ✅ Yes | Deep sentence understanding |