

Similarity score using AI

Monday, 27 January 2025

4:52 PM

```
Import numpy as np
From open ai import OpenAI
```

```
Client = OpenAI()
```

```
Response = client.embedding.create(
    Input=["potato","rhubarb"],
    Model="text-embedding-ada-002"
)
```

```
Potato = response.data[0].embedding
```

```
Rhubarb = response.data[1].embedding
```

```
SimScore = np.dot(potato,rhubarb)
```

=====

Without using AI with transformers:

```
from sentence_transformers import SentenceTransformer
import numpy as np
```

```
def calculate_similarity(text1: str, text2: str) -> float:
```

```
    """
```

```
    Calculate the cosine similarity between two texts using Sentence Transformers.
```

```
    :param text1: First text input.
```

```
    :param text2: Second text input.
```

```
    :return: Cosine similarity score between the two text embeddings.
```

```
    """
```

```
    # Load the pre-trained model
```

```
    model = SentenceTransformer('all-MiniLM-L6-v2')
```

```
    # Generate embeddings for the input texts
```

```
    embeddings = model.encode([text1, text2])
```

```
    # Extract individual embeddings
```

```
    embedding1, embedding2 = embeddings[0], embeddings[1]
```

```
    # Calculate cosine similarity
```

```
    similarity = np.dot(embedding1, embedding2) / /
```

```
similarity = np.dot(embedding1, embedding2) / (  
    np.linalg.norm(embedding1) * np.linalg.norm(embedding2)  
)
```

```
return similarity
```

```
if __name__ == "__main__":
```

```
    # Example usage: Test the method with two sample texts
```

```
    text1 = "potato"
```

```
    text2 = "rhubarb"
```

```
    # Call the similarity calculation method
```

```
    similarity_score = calculate_similarity(text1, text2)
```

```
    # Print the result
```

```
    print(f"Cosine Similarity between '{text1}' and '{text2}': {similarity_score:.4f}")
```