EX NO. : 1

DATE:

Implement Data similarity measures using Python.

AIM:

To implement data similarity measures using python.

ALGORITHM:

1. Import the necessary libraries.
2. Define the function.
3. Get the data and implement in the formula.
4. Print the output of the data.

PROGRAM:

Numeric Data:-

1. **Euclidean Distance:**

```
import numpy as np

def euclidean_distance(x, y):

    return np.sqrt(np.sum((x - y) ** 2))

# Example usage:

data1 = np.array([1, 2, 3])

data2 = np.array([4, 5, 6])

distance = euclidean_distance(data1, data2)

      print(f"Euclidean Distance: {distance}")
```

2. Cosine Similarity:

```
from sklearn.metrics.pairwise import cosine_similarity
data1 = np.array([1, 2, 3])
data2 = np.array([4, 5, 6])
```

```
# Reshape data to be 2D arrays
data1 = data1.reshape(1, -1)
data2 = data2.reshape(1, -1)

cosine_sim = cosine_similarity(data1, data2)
    print(f"Cosine Similarity: {cosine_sim[0][0]}")
```

Text Data:-

3. Jaccard Similarity:

```
def jaccard_similarity(set1, set2):
        intersection = len(set1.intersection(set2))
        union = len(set1.union(set2))
        return intersection / union

# Example usage:
text1 = set("hello world")
text2 = set("world hello")

similarity = jaccard_similarity(text1, text2)
print(f"Jaccard Similarity: {similarity}")
```

4. Levenshtein Distance(Edit Distance):

```
def levenshtein_distance(str1, str2):
    if len(str1) > len(str2):
        str1, str2 = str2, str1


    distances = range(len(str1) + 1)
    for index2, char2 in enumerate(str2):
        new_distances = [index2 + 1]
        for index1, char1 in enumerate(str1):
            if char1 == char2:
                new_distances.append(distances[index1])
                        else:
                 new_distances.append(1 + min((distances[index1], distances[index1 + 1],
              new_distances[-1])))
        distances = new_distances
    return distances[-1]
```

```
# Example usage:
word1 = "kitten"
word2 = "sitting"
distance = levenshtein_distance(word1, word2)
    print(f"Levenshtein Distance: {distance}")
```

OUTPUT:

```
1.  Euclidean Distance: 5.196152422706632
2.  Cosine Similarity: 0.9746318461970762
3.  Jaccard Similarity: 1.0
4.  Levenshtein Distance: 3
```

RESULT:

Thus the implementation of data similarity measures using python was executed successfully.