شرح خوارزمية (Hamming Similarity)

عائشة عبد السلام الحِدَابِي
20-16-0149
CS4

نظرة عامة عن مفهوم الخوارزمية:

تعريفها:

هي طريقة لقياس التشابه بين سلسلتين ثنائيتين (binary strings) من نفس الطول. تستخدم هذه الخوارزمية لحساب عدد البتات المختلفة بين السلسلتين، وبالتالي تقييم درجة التشابه بينهما.

يعتبر مقياس التشابه هامينغ بسيطًا وفعالًا للسلاسل الثنائية، مثل البيانات المشفرة أو العناوين البريدية أو البيانات الرقمية الأخرى التي تستخدم تمثيلًا ثنائيًا. تعتمد الخوارزمية على مبدأ حساب البتات المختلفة بين السلسلتين بوضعهما جنبًا إلى جنب ومقارنة القيمة الثنائية لكل بت.

الخوارزمية تعمل كالتالي:

1. تحديد سلسلتين ثنائيتين متساويتين في الطول.
2. مقارنة البتات المتطابقة بين السلسلتين. إذا كانت القيمة الثنائية للبت في كلا السلسلتين متطابقة، فإنها تُعتبر متشابهة، وإذا كانت مختلفة، فإنها تعتبر غير متشابهة.
3. بعد الانتهاء من مقارنة جميع البتات يتم حساب عدد البتات المختلفة بين السلسلتين.

نتيجة الخوارزمية هي عدد البتات المختلفة بين السلسلتين، ويمكن استخدامها لقياس الشبه أو التشابه بين السلسلتين الثنائية. يتم استخدام هذه الخوارزمية في العديد من التطبيقات، مثل إدارة الأخطاء، والتعرف على الأنماط، وتحقق التشفير، وغيرها.

مثال:

لنفترض أن لدينا سلسلتين ثنائيتين كما يلي:

سلسلة 1: 101010

سلسلة 2: 111000

لحساب مقياس التشابه حسب خوارزمية Hamming Similarity، نقوم بمقارنة البتات المتطابقة في كلا السلسلتين وحساب عدد البتات المختلفة بينهما.

مقارنة البتات:

- البت الأول: القيمة في السلسلة 1 هي 1 وفي السلسلة 2 هي 1 (متطابقة).
- البت الثاني: القيمة في السلسلة 1 هي 0 وفي السلسلة 2 هي 1 (غير متطابقة).
- البت الثالث: القيمة في السلسلة 1 هي 1 وفي السلسلة 2 هي 1 (متطابقة).
- البت الرابع: القيمة في السلسلة 1 هي 0 وفي السلسلة 2 هي 0 (متطابقة).
- البت الخامس: القيمة في السلسلة 1 هي 1 وفي السلسلة 2 هي 0 (غير متطابقة).
- البت السادس: القيمة في السلسلة 1 هي 0 وفي السلسلة 2 هي 0 (متطابقة).

بعد إجراء المقارنة، نجد أن هناك 2 بتات مختلفة بين السلسلتين. وبالتالي، مقياس التشابه بينهما وفقًا لخوارزمية Hamming Similarity هو 2.

هذا هو مثال بسيط لتوضيح كيفية استخدام خوارزمية Hamming Similarity لحساب التشابه بين سلسلتين ثنائيتين. يمكن تطبيق نفس العملية على سلاسل أطول وأكثر تعقيدًا.

الكود البرمجي:

```
def hamming_similarity(string1, string2):
    if len(string1) != len(string2):
        raise ValueError("Both strings must have the same length.")
    hamming_distance = 0
    for i in range(len(string1)):
        if string1[i] != string2[i]:
            hamming_distance += 1
    similarity = len(string1) - hamming_distance
    return similarity
# Exaample usage
string1 = "101010"
string2 = "111000"
similarity_score = hamming_similarity(string1, string2)
print("Hamming similarity score:", similarity_score)
```