1458. Max Dot Product of Two Subsequences

Given two arrays `nums1` and `nums2`.

Return the maximum dot product between **non-empty** subsequences of nums1 and nums2 with the same length.

A subsequence of a array is a new array which is formed from the original array by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters. (ie, `[2,3,5]` is a subsequence of `[1,2,3,4,5]` while `[1,5,3]` is not).

Example 1:

```
Input: nums1 = [2,1,-2,5], nums2 = [3,0,-6]
Output: 18
Explanation: Take subsequence [2,-2] from nums1 and subsequence [3,-6]
from nums2.
Their dot product is (2*3 + (-2)*(-6)) = 18.
```

Example 2:

```
Input: nums1 = [3,-2], nums2 = [2,-6,7]
Output: 21
Explanation: Take subsequence [3] from nums1 and subsequence [7] from
nums2.
Their dot product is (3*7) = 21.
```

Example 3:

```
Input: nums1 = [-1,-1], nums2 = [1,1]
Output: -1
Explanation: Take subsequence [-1] from nums1 and subsequence [1] from
nums2.
Their dot product is -1.
```

```
class Solution:
    def maxDotProduct(self, nums1: List[int], nums2: List[int]) -> int:
        if (max(nums1) < 0 and min(nums2) > 0):
            return max(nums1) * min(nums2)
        if (max(nums2) < 0 and min(nums1) > 0):
            return min(nums1) * max(nums2)
        m = len(nums1)
        n = len(nums2)
        dp = [[0]*(m+1) for _ in range(n+1)]
        for i in range(1,n+1):
            for j in range(1,m+1):
                dp[i][j] = max(dp[i-1][j-1]+nums1[j-1]*nums2[i-1],dp[i][j-
1],dp[i-1][j])
        return dp[-1][-1]
```