Description | Accepted | Editorial | Submissions | Solutions

1570. Dot Product of Two Sparse Vectors Premium

Solved

Medium | Topics | Companies | Hint

Given two sparse vectors, compute their dot product.

Implement class `SparseVector`:

- `SparseVector(nums)` Initializes the object with the vector `nums`
- `dotProduct(vec)` Compute the dot product between the instance of *SparseVector* and `vec`

A **sparse vector** is a vector that has mostly zero values, you should store the sparse vector **efficiently** and compute the dot product between two *SparseVector*.

Follow up: What if only one of the vectors is sparse?

Example 1:

```
Input: nums1 = [1,0,0,2,3], nums2 = [0,3,0,4,0]
Output: 8
Explanation: v1 = SparseVector(nums1) , v2 = SparseVector(nums2)
v1.dotProduct(v2) = 1*0 + 0*3 + 0*0 + 2*4 + 3*0 = 8
```

Example 2:

```
Input: nums1 = [0,1,0,0,0], nums2 = [0,0,0,0,2]
Output: 0
Explanation: v1 = SparseVector(nums1) , v2 = SparseVector(nums2)
v1.dotProduct(v2) = 0*0 + 1*0 + 0*0 + 0*0 + 0*2 = 0
```

Example 3:

```
Input: nums1 = [0,1,0,0,2,0,0], nums2 = [1,0,0,0,3,0,4]
Output: 6
```

Constraints:

- `n == nums1.length == nums2.length`
- `1 <= n <= 10^5`
- `0 <= nums1[i], nums2[i] <= 100`

Seen this question in a real interview before? 1/5

Yes | No

Accepted 253.3K | Submissions 281.9K | Acceptance Rate 89.9%

Topics

Companies

Hint 1

Discussion (15)

1.2K | 15

Code

Python3 • Auto

```python
class SparseVector:
    def __init__(self, nums: List[int]):
        self.dict = {}
        for i, n in enumerate(nums):
            if n != 0:
                self.dict[i] = n


    # Return the dotProduct of two sparse vectors
    def dotProduct(self, vec: 'SparseVector') -> int:
        result = 0
        if not self.dict or not vec.dict:
            return 0
        low, high = min(self.dict.keys()), max(self.dict.keys())
        for i in range(low, high+1):
            if (i in self.dict) and (i in vec.dict):
                result += self.dict[i] * vec.dict[i]
        return result


# Your SparseVector object will be instantiated and called as such:
# v1 = SparseVector(nums1)
# v2 = SparseVector(nums2)
# ans = v1.dotProduct(v2)
```

Saved Ln 11, Col 19

Testcase | Test Result

Accepted Runtime: 48 ms

• Case 1 • Case 2 • Case 3

Input