410. Split Array Largest Sum

Given an array `nums` which consists of non-negative integers and an integer `m`, you can split the array into `m` non-empty continuous subarrays.

Write an algorithm to minimize the largest sum among these `m` subarrays.

Example 1:

```
Input: nums = [7,2,5,10,8], m = 2
Output: 18
Explanation:
There are four ways to split nums into two subarrays.
The best way is to split it into [7,2,5] and [10,8],
where the largest sum among the two subarrays is only 18.
```

Example 2:

```
Input: nums = [1,2,3,4,5], m = 2
Output: 9
```

Example 3:

```
Input: nums = [1,4,4], m = 3
Output: 4
```

Constraints:

- `1 <= nums.length <= 1000`
- `0 <= nums[i] <= 10<sup>6</sup>`
- `1 <= m <= min(50, nums.length)`

```
class Solution:
    def splitArray(self, A: List[int], M: int) -> int:
        N = len(A)
        if N==M:
            return max(A)
        if N<M:
            return -1
        lo = max(A)
        hi = sum(A)
        ans = -1
        while lo<=hi:
```

```python
            mid = (lo+hi)//2

            if self.isValid(A,mid,M):
                ans = mid
                hi = mid-1
            else:
                lo = mid+1
        return ans


    def isValid(self,A,pages,M):
        tempSum = 0
        students = 1

        for i in range(len(A)):
            tempSum = tempSum+A[i]
            if tempSum>pages:
                students+=1
                tempSum = A[i]

        if students>M:
            return False
        else:
            return True
```