1182. Shortest Distance to Target Color

My Submissions (/contest/biweekly-contest-8/problems/shortest-distance-to-target-color/submissions/)

Back to Contest (/contest/biweekly-contest-8/)

You are given an array `colors`, in which there are three colors: `1`, `2` and `3`.

You are also given some queries. Each query consists of two integers `i` and `c`, return the shortest distance between the given index `i` and the target color `c`. If there is no solution return `-1`.

User Accepted:	577
User Tried:	740
Total Accepted:	586
Total Submissions:	1362
Difficulty:	Medium

Example 1:

```
Input: colors = [1,1,2,1,3,2,2,3,3], queries = [[1,3],[2,2],[6
Output: [3,0,3]
Explanation:
The nearest 3 from index 1 is at index 4 (3 steps away).
The nearest 2 from index 2 is at index 2 itself (0 steps away)
The nearest 1 from index 6 is at index 3 (3 steps away).
```

Example 2:

```
Input: colors = [1,2], queries = [[0,3]]
Output: [-1]
Explanation: There is no 3 in the array.
```

Constraints:

- `1 <= colors.length <= 5*10^4`
- `1 <= colors[i] <= 3`
- `1 <= queries.length <= 5*10^4`
- `queries[i].length == 2`
- `0 <= queries[i][0] < colors.length`
- `1 <= queries[i][1] <= 3`

Discuss (https://leetcode.com/problems/shortest-distance-to-target-color/discuss)

Python3

```
class Solution:
    # 2032 ms, using bisect
    def shortestDistanceColor(self, colors: List[int], queries: List[List[int]]) ->
List[int]:
        D = collections.defaultdict(list)
        for i, v in enumerate(colors):
```

```
            D[v].append(i)
        print(D)
        ans = []
        for i, v in queries:
            if v not in D:
                ans.append(-1)
                continue
            index = bisect.bisect(D[v], i)
            temp1 = D[v][index] if index < len(D[v]) else float("inf")
            temp2 = D[v][index - 1] if index>0 else float("inf")
            ans.append(min(abs(i - temp1), abs(i - temp2)))
        return ans

    # done in contest 2100 ms, search left and right, naive
    def shortestDistanceColor1(self, colors: List[int], queries: List[List[int]]) ->
List[int]:

        def nearest_left(index, val):
            for i in reversed(range(0, index +1)):
                if colors[i] == val:
                    return index - i
            return -1

        def nearest_right(index, val):
            for i in range(index, len(colors)):
                if colors[i] == val:
                    return i - index
            return -1

        ans = []
        D = {}
        for i, val in queries:
            if (i, val) in D:
                ans.append(D[(i,val)])
                continue
            left = nearest_left(i, val)
            right = nearest_right(i, val)
            if -1 in (left,right):
                rv = max(left, right)
            else:
                rv = min(left, right)
            ans.append(rv)
            D[(i,val)] = rv
        return ans
```

☐ **Custom Testcase** Run Code Submit