Problem : <https://leetcode.com/problems/the-number-of-good-subsets/submissions/>

Approach :

-> At most 1024 subsets have to be checked for each number because there are only 10 primes till 30.

-> Just bcoz nums[i] is at most 30, we can actually iterate over **count** of all numbers rather than the whole 10^5 numbers in array itself.

-> Store frequence of each number in cnts[] array of size 30.

-> Now for i from 1...31:

-> If number is 1 : continue

-> Else If number is div by 4 , 9 or 25 : continue (as this number cant be part of any subset as it has multiple occurrences of a prime number)

-> Else

Generate bitmask for the number . E.g If number contains 2 and 3 and 5 as prime factors then bitmask = 0000000111 , as the bits from (lsb to msb) represent prime numbers from 2 till 29.

Now for all the bitmask generated for numbers behind ‘i’ , check is there is any bitmask such that older bitmask ANDED current number bitmask ==0 , if yes then this subset is also possible, **name it generated bitmask.**

**dp[generated bitmask] = dp[generated bitmask] + (dp[older bitmask]\*cnt[current number])**

where dp[k] means number of occurrences of bitmask valued ‘k’.So note that every bitmask can occur multiple times as same subset can occur multiple times as there are many duplicates.

**-> 1s have to be dealt with separately.** All combinations of 1s can be multiplied with all above generated combinations to give final answer.

Code: <https://leetcode.com/submissions/detail/549863718/>