Better:

def findMissingRepeatingNumbers(a: [int]) -> [int]:

    f=len(a)

    b=list(set(a))

    mis=((f\*(f+1))//2)-sum(b)

    rep=(sum(a)+mis)-(f\*(f+1)//2)

    return [rep,mis]

Optimal:

def findMissingRepeatingNumbers(a: [int]) -> [int]:

    n=len(a)

    xr=0

    for i in range(n):

        xr=xr^a[i]

        xr=xr^(i+1)#xor all numbers in array with n numbers

    #xr will have mis^rep

    #Find the position where the first bit in mis and rep differ

    num=xr&~(xr-1)  #bit will always have value zero and 1

    zer,one=0,0#we are separating numbers in category

    for i in range(n):

        if (a[i]&num)==0:

            zer^=a[i]

        else:

            one^=a[i]

    for i in range(1,n+1):

        if (i&num)==0:

            zer^=i

        else:

            one^=i

    #After this step we have mis and rep in either zer and one

    count=0

    for i in a:

        if i==zer:

            count+=1

    if count==2:

        return [zer,one]

    else:

        return [one,zer]