Brute force:

def triplet(n: int, arr: [int]) -> [[int]]:

    l=set()

    for i in range(n):

        for j in range(i+1,n):

            for k in range(j+1,n):

                if arr[i]+arr[j]+arr[k]==0:

                    temp=[arr[i],arr[j],arr[k]]

                    temp.sort()

                    l.add(tuple(temp)

    ans = [list(item) for item in l]

    return ans

Better:

def triplet(n: int, arr: [int]) -> [[int]]:

    #Better Solution optimized to o(n2)

    l=set()

    for i in range(n):

        d={}

        for j in range(i+1,n):

            s=-(arr[i]+arr[j])#The remaining elemnt to make it zero

            if s in d:

                temp=[arr[i],arr[j],s]

                temp.sort()

                l.add(tuple(temp))

            d[arr[j]]=j #This is in the end so  repetition is not allowed of the positions

    ans = [list(item) for item in l]

    return ans

Optimal:

def triplet(n: int, arr: [int]) -> [[int]]:

    arr.sort()

    i=0

    l=[]

    for i in range(n):

        if(i>0 and arr[i]==arr[i-1]):#Fixing first position

            continue#To avoid duplicates

        left=i+1#implementing 2 sum

        right=n-1

        while(left<right):

            s=arr[left]+arr[right]+arr[i]

            if s==0:

                l.append([arr[i],arr[left],arr[right]])

                right-=1

                left+=1

                while left<right and arr[left-1]==arr[left]:

                    left+=1#avoid repetion

                while left<right and arr[right+1]==arr[right]:

                    right-=1    #avoid repetitionS

            elif s<0:

                left+=1

            else:

                right-=1

    return l