import os

import os.path

import re

from itertools import islice

import math

import numpy  as np

class rgb\_choose:

    #将新基本点加入样本

    #对所有目标求对应样本

    #目标与相似度偏差最小的样本一一对应，求色差

    #22个原样本

    def pic1():

        colors\_add =[

            [95,47,153],

            [62,199,80],

            [32,85,204],

            [172,118,86],

            [174,118,213],

            [38,111,77],

            [40,171,226],

            [197,27,164],

            [35,231,199],

            [162,216,240]

        ]

        #10个新样本

        f\_sum=[]

        #存放所有效益

        for i in range(len(colors\_add)):

            colors = [[0,0,0],

                [255,255, 255],

                [255, 0, 0],

                [246, 232, 9],

                [72, 176, 64],

                [27, 115, 186],

                [53, 118, 84],

                [244, 181, 208],

                [255, 145, 0],

                [177, 125, 85],

                [92, 59, 144],

                [11, 222, 222],

                [228, 0, 130],

                [225, 218, 32],

                [118, 238, 0],

                [17, 168, 226],

                [255, 110, 0],

                [201, 202, 202],

                [255, 249, 177],

                [179, 226, 242],

                [249, 225, 214],

                [186, 149, 195]

            ]

            colors.extend(colors\_add[0:i])

            pic2=open('D:/twinkle/桌面/数学建模华中杯/附件/附件2：图像1颜色列表.txt', mode='r')

            next(pic2)

            #跳过标题

            dic=dict()

            #创建空字典，用于存储目标

            for line in pic2:

                '''按行读取'''

                k,v =line.split(',',1)  # 以','为分隔符分割字符串

                dic[k] = v[0:-1]  #去掉字符串末尾的换行符

            dic=list(dic.values())

            # print(dic)

            rgb\_sum=[]

            #创建空列表

            for i in range(len(dic)):

                rgb=re.findall(r'\d+',dic[i])

                #构建循环，正则表达提取rgb对应数值并添加到列表

                rgb = [int(x) for x in rgb]

                #将列表中的字符串转为数字

                rgb\_sum.append(rgb)

                pic2.close()

                # print (rgb)

            # print (rgb\_sum[1][1])

            # print(pic2.read())

            '''计算三维向量在空间中的距离,衡量样本与目标之间的差异'''

            solver1 =[]

            #创建一个空列表，用于储存结果

            for j in range(len(rgb\_sum)):

                vector2 = np.array(list(rgb\_sum[j]))

                result =[]

                for i in range(len(colors)):

                    vector1 = np.array(colors[i])

                    op = np.linalg.norm(vector1-vector2)

                    #欧式距离范式

                    result.append(op)

                solver1.append(result.index(min(result)))

            delta\_c\_sum = 0.0

            #存放色差和

            for i in range(len(rgb\_sum)):

                def ColourDistance(rgb\_1, rgb\_2):

                    '''计算LAB颜色空间中的色差，为简化计算，在RGB空间上直接算出加权欧式距离'''

                    R\_1,G\_1,B\_1 = rgb\_1

                    R\_2,G\_2,B\_2 = rgb\_2

                    rmean = (R\_1 +R\_2 ) / 2

                    R = R\_1 - R\_2

                    G = G\_1 -G\_2

                    B = B\_1 - B\_2

                    return math.sqrt((2+rmean/256)\*(R\*\*2)+4\*(G\*\*2)+(2+(255-rmean)/256)\*(B\*\*2))

                delta\_c=ColourDistance(colors[solver1[i]],rgb\_sum[i])

                #参数为  替换目标的样本 样本

                delta\_c\_sum += delta\_c

            f =   - (delta\_c\_sum / i) + (1 / i)

            f = abs(f)

            #衡量效益

            print(f)

            f\_sum.append(f)

        print ('='\*60)

        print ('对于图像1，考虑成本与表现效果，应该添加的颜色种数为：',f\_sum.index(min(f\_sum))+1)

        print ('应该添加的几种颜色的值为',colors\_add[0:f\_sum.index(min(f\_sum))+1])

        print('='\*60)

    def pic2():

        colors\_add =[

            [85,33,148],

            [173,128,79],

            [65,199,95],

            [178,125,205],

            [40,109,73],

            [47,101,202],

            [216,33,134],

            [44,165,215],

            [32,220,171],

            [240,168,217]

        ]

        #10个新样本

        f\_sum=[]

        #存放所有效益

        for i in range(len(colors\_add)):

            colors = [[0,0,0],

                [255,255, 255],

                [255, 0, 0],

                [246, 232, 9],

                [72, 176, 64],

                [27, 115, 186],

                [53, 118, 84],

                [244, 181, 208],

                [255, 145, 0],

                [177, 125, 85],

                [92, 59, 144],

                [11, 222, 222],

                [228, 0, 130],

                [225, 218, 32],

                [118, 238, 0],

                [17, 168, 226],

                [255, 110, 0],

                [201, 202, 202],

                [255, 249, 177],

                [179, 226, 242],

                [249, 225, 214],

                [186, 149, 195]

            ]

            colors.extend(colors\_add[0:i])

            pic2=open('D:/twinkle/桌面/数学建模华中杯/附件/附件3：图像2颜色列表.txt', mode='r')

            next(pic2)

            #跳过标题

            dic=dict()

            #创建空字典，用于存储目标

            for line in pic2:

                '''按行读取'''

                k,v =line.split(',',1)  # 以','为分隔符分割字符串

                dic[k] = v[0:-1]  #去掉字符串末尾的换行符

            dic=list(dic.values())

            # print(dic)

            rgb\_sum=[]

            #创建空列表

            for i in range(len(dic)):

                rgb=re.findall(r'\d+',dic[i])

                #构建循环，正则表达提取rgb对应数值并添加到列表

                rgb = [int(x) for x in rgb]

                #将列表中的字符串转为数字

                rgb\_sum.append(rgb)

                pic2.close()

                # print (rgb)

            # print (rgb\_sum[1][1])

            # print(pic2.read())

            '''计算三维向量在空间中的距离,衡量样本与目标之间的差异'''

            solver1 =[]

            #创建一个空列表，用于储存结果

            for j in range(len(rgb\_sum)):

                vector2 = np.array(list(rgb\_sum[j]))

                result =[]

                for i in range(len(colors)):

                    vector1 = np.array(colors[i])

                    op = np.linalg.norm(vector1-vector2)

                    #欧式距离范式

                    result.append(op)

                solver1.append(result.index(min(result)))

            delta\_c\_sum = 0.0

            #存放色差和

            for i in range(len(rgb\_sum)):

                def ColourDistance(rgb\_1, rgb\_2):

                    '''计算LAB颜色空间中的色差，为简化计算，在RGB空间上直接算出加权欧式距离'''

                    R\_1,G\_1,B\_1 = rgb\_1

                    R\_2,G\_2,B\_2 = rgb\_2

                    rmean = (R\_1 +R\_2 ) / 2

                    R = R\_1 - R\_2

                    G = G\_1 -G\_2

                    B = B\_1 - B\_2

                    return math.sqrt((2+rmean/256)\*(R\*\*2)+4\*(G\*\*2)+(2+(255-rmean)/256)\*(B\*\*2))

                delta\_c=ColourDistance(colors[solver1[i]],rgb\_sum[i])

                #参数为  替换目标的样本 样本

                delta\_c\_sum += delta\_c

            f =   - (delta\_c\_sum / i) + (1 / i)

            f = abs(f)

            #衡量效益

            print(f)

            f\_sum.append(f)

        print ('='\*60)

        print ('对于图像2，考虑成本与表现效果，应该添加的颜色种数为：',f\_sum.index(min(f\_sum))+1)

        print ('应该添加的几种颜色的值为',colors\_add[0:f\_sum.index(min(f\_sum))+1])

    pic1()

    pic2()