Dominant Color Extraction

We can segment image into region which have similar visual appearance,this technique is used to image compression as well.



This project is done By k-means Algorithm

1.import cv2 #(to read the image)

:

**import** **matplotlib.pyplot** **as** **plt**

2. im=cv2.imread("C:/Users/HP/Downloads/bird.jpg")

# to read image

original\_shape=im.shape

# to get shape of image

3.maltplotlib give output of image in BGR format ,to convert it into RGB

im=cv2.cvtColor(im,cv2.COLOR\_BGR2RGB)

plt.imshow(im)

4. *#faltten each channel*

all\_pixels=im.reshape((-1,3))

print(all\_pixels.shape)

5. **from** **sklearn.cluster** **import** KMeans

#importing Kmeans

6. dominant\_colors=3

km=KMeans(dominant\_colors)

km.fit(all\_pixels)

# through this we are only suppose to get only three dominant color

7. centers=km.cluster\_centers\_

#to get RGB value for the color

8. **import** **numpy** **as** **np**

**# to convert the color into numpy**

**9.**   
i=1

plt.figure(0,figsize=(4,2)) #to declare the size of image

colors=[]

**for** each\_col **in** centers:

plt.subplot(1,10,i) #to create subplot

i+=1

colors.append(each\_col)

*#color swatch*

a=np.zeros((100,100,3),dtype='uint8') #to create matrix of 100\*100\*3

a[:,:,:]=each\_col

plt.imshow(a)

plt.show()

10. new\_img=np.zeros((142\*355,3),dtype='uint8')

print(new\_img.shape)

# creating new image of same size as of original image

11. km.labels\_

# through this we are able to see which part belongs to which segment

12. **for** ix **in** range(new\_img.shape[0]):

new\_img[ix]=colors[km.labels\_[ix]]

new\_img=new\_img.reshape((original\_shape))

plt.imshow(new\_img)

# new image is reshaped into original