Machine Learning

Assignment 1

Instructor: Dr.Wajahat Hussain TA: Sunaina Ajmal

Submitted by: Misbah Naeem CMS ID:362727

Problem Statement:

In this assignment, I implement K-Nearest Neighbour. A file of dataset was given as data.mat whoich conatnin iages of 50 random celbrites .I need to upload my images on colab and using nearest neighbour classifier I need to use to which celerbity I resemble the most.

Import Nrequired packages in colab.Mount google drive and import dataset file in colab.Output shows that drive is successfully mounted and dataet file is read successfully.

Code:

```
#Enter your path of dataset from google drive
import scipy.io as sio
GOOGLE_COLAB = True
path = ""
if GOOGLE_COLAB:
    from google.colab import drive, files
    drive.mount('/content/drive/')
    path = "/content/drive/My Drive/"

dataset = path + "data.mat"

#Enter path of your test image
test_image=path+"test.jpg"
mat_contents = sio.loadmat(dataset)
mat_contents
images = mat_contents['images']
label = mat_contents['C']
images.shape
```

Output:

```
Drive already mounted at /content/drive/; to attempt to forcibly remount, call drive.mount("/content/drive/", force_remount
(50, 3072)
```

Import Numpy which is used for vast number of purposes specially to perform Mathematics functions. Reshape our dataset images and display their shape on screen. Display some random image from dataset. Images are of low resolution (32, 32, 3).

Code:

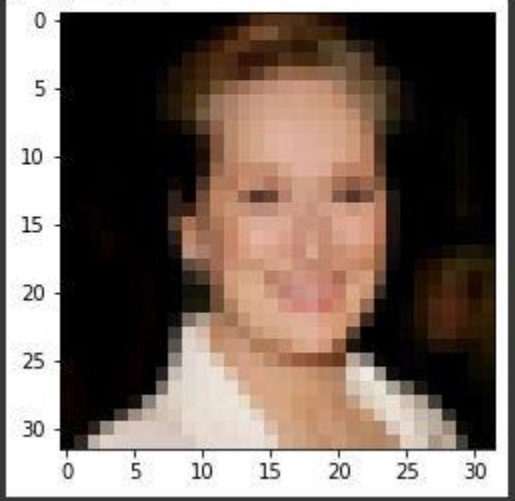
```
import numpy as np
images= np.transpose(images)
images.shape
im = np.reshape(images, [ 32, 32, 3, -1], order="F")
im.shape
```

```
(32, 32, 3, 50)
```

```
from matplotlib import pyplot as plt
#import cv2

plt.imshow(im[:,:,:,22])
print('final diension', im[:,:,:,0].shape)
a= im[:,:,:,22]
plt.imshow(a)
a.shape
```

Output:

```
final dimension (32, 32, 3)
(32, 32, 3)
```



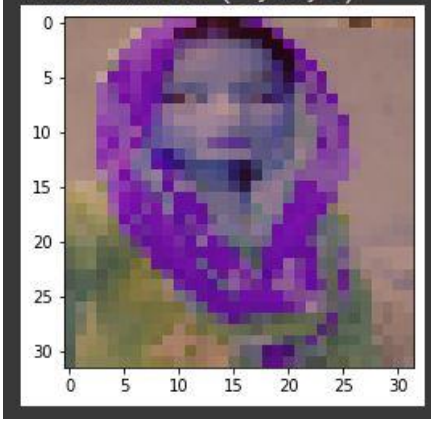
Import necessary packages like cv2, Numpy and math. Read an image from drive which I will compare with the dataset images to find KNN. My original image has different dimension then my dataset images so I need to resize and reshape my image to calculate distance. So I resize image to (32, 32, 3) dimension. My output of this block shows final dimension and output image.

Code:

```
from scipy import misc
import cv2
from math import sqrt,pow
from numpy import ndarray
import numpy
#Read your image here
from PIL import Image
from google.colab.patches import cv2_imshow

me=cv2.imread('/content/drive/MyDrive/iqra.jpg')
print('original dimension',me.shape)
plt.imshow(me)
############################
#Resize your image
width=32
height=32
dim=(width,height)
resized = cv2.resize(me,dim)
print('resized dimension',resized.shape)
plt.imshow(resized)
############################
#Reshape your image as we reshape the image of dataset
ima = np.reshape(resized, [32,32,3])
plt.imshow(ima)
print('final dimension',ima.shape)
############################
```

Output:

```
original dimension (948, 508, 3)
resized dimension (32, 32, 3)
final dimension (32, 32, 3)
```



Now dataset and test image is ready to use so I made a function named “dist” to calculate distance between two images.image1 keyword contain my image and I know that I have total 50 images in my dataset so I loop over dataset and pick one by one all images from there and find their distances, assign all the distances to an array “distance”. Output shows all 50 values of distances of my image with the dataset images.

Code:

```
#Calculate Eucleadian distance between your image and dataset
def dist(image1,image2):
  X=numpy.sum((image1 - image2) **2)
  Y=sqrt(X)
  return Y

image1 = ima #ima is my image
distance=50*[0]
for i in range(0,50):
  distance[i]=dist(ima , im[:,:,:,i])
  print("distance",i,distance[i])
```

Output:

```
distance 0 572.7076042798803
distance 1 573.9581866303503
distance 2 578.1660661090376
distance 3 578.2897197772065
distance 4 573.5372699310831
distance 5 540.4932932053829
distance 6 569.8306766049017
distance 7 582.1941257003543
distance 8 568.5120930991706
distance 9 563.5432547728701
distance 10 567.7270470921744
distance 11 583.2075102397088
distance 12 574.1149710641589
distance 13 579.6524820959538
distance 14 576.6922923015358
distance 15 567.9735909353533
distance 16 562.643759407318
distance 17 572.5329684830385
distance 18 566.5342002033063
distance 19 568.9666071044943
distance 20 577.7897541493792
distance 21 569.3346994519129
distance 22 541.8542239385055
distance 23 564.6255396278139
distance 24 550.6659604515246
distance 25 575.3042673229533
distance 26 570.4173910392284
distance 27 576.7009623713143
distance 28 573.320154887302
distance 29 563.1456294778466
distance 30 576.4971812593709
distance 31 571.9300656548841
distance 32 553.060575344148
distance 33 571.6808550231501
distance 34 583.6351600100871
distance 35 576.7538469745997
distance 36 571.5680186994371
distance 37 570.6040658810625
distance 38 567.6416122871896
distance 39 565.3229165706977
distance 40 564.7495019918123
distance 41 557.5733135651311
distance 42 562.4704436679318
distance 43 559.0751291195129
distance 44 568.7363185167623
distance 45 549.0400713973435
distance 46 573.1151716714538
distance 47 576.1423435228485
distance 48 569.4708069778468
distance 49 571.0061295642981
```

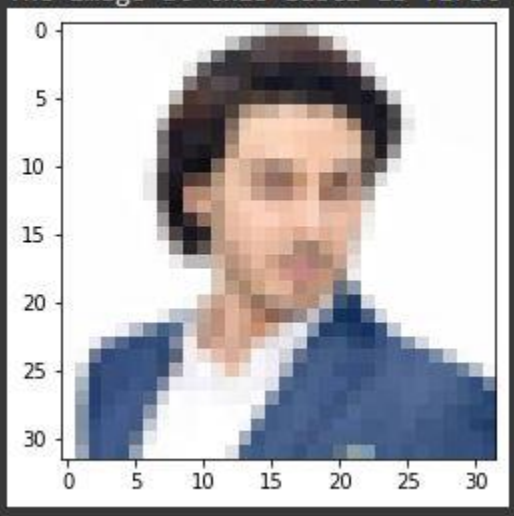
At this point, I have an array of distances from all the images .Now I find the minimum value of distance and which image from dataset has this minimum distance and print that image on screen because this image is my first nearest neighbor. First I use **min** keyword to find lowest distance value, in this way I get lowest value at the first then using a loop I find that index which has this value and print image at that index as my output. Output shows lowest value, its index in the array and image at this index.

Code:

```
# Write code for 1 NN
#Find min distance
#Fine at which point min value exists

####### Your code here #######
#print('distance:\n',distance)
print("minimum distance:  ",min(distance))

for i in range(0,50):
  if distance[i]==min(distance):
    print("The image at this label is first nearest neighbour: ", i)
    plt.imshow(im[:,:,:,i])


##############################
```

Output:

```
minimum distance:   540.4932932053829
The image at this label is first nearest neighbour:  5
```



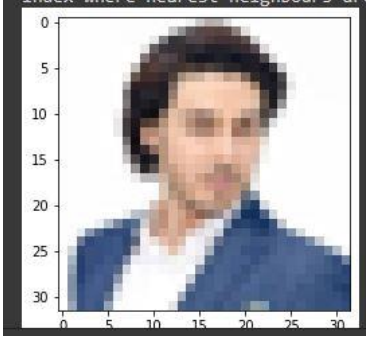
In previous part I have find 1 nearest neighbor. Now I will find first 3 nearest neighbor. So I use sort keyword to sort y array in ascending order, first three values are the lowest distance values. Using loop I extract these three values and then in another loop I find these three lowest values from original distance array and their indexes to print required nearest neighbor images. Output shows original array, sorted array, first three minimum values of distances, their indexes and images at these indexes.

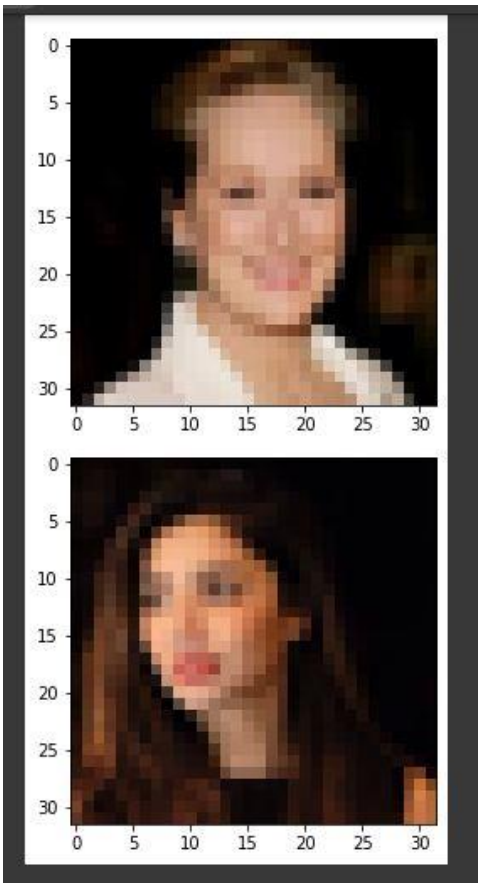
Code:

```
#3 NN
#Write code for 3 NN
#Find 3 min distances
#Find their instances
#hint: Use for Loop
x=50*[0]
####### Your code here #######
print("original distance:\n",distance)
sorted_dist=sorted(distance)
print("sorted distance array :\n",sorted_dist)
for i in range (0,3):
  x[i]=sorted_dist[i]
  print("first 3 minimum values",x[i])
for i in range(len(x)):
  for j in range(len(y)):
    if x[i] == distance[j]:
      print("index where nearest neighbours are placed:", j)
      plt.subplots()
      plt.imshow(im[:,:,:,j])
##############################
```

Output:

```
original distance:
 [572.7076042798803, 573.9581866303503, 578.1660661090376, 578.2897197772065, 573.5372699310831, 540.4932932053829,
sorted distance array :
 [540.4932932053829, 541.8542239385055, 549.0400713973435, 550.6659604515246, 553.060575344148, 557.5733135651311, 5
first 3 minimum values 540.4932932053829
first 3 minimum values 541.8542239385055
first 3 minimum values 549.0400713973435
index where nearest neighbours are placed: 5
index where nearest neighbours are placed: 22
index where nearest neighbours are placed: 45
```





As I already have sorted array so using for loop I took 5 lowest values and store them in another array .using another array I match the values from original distance array to find required indexes and print corresponding images. Output shows first five lowest values of distances, their corresponding indexes and images

Code:

```
#Write code for 5 NN
#Find 5 min distances
#Find their instances
x=50*[0]

####### Your code here #######
for i in range (0,5):
  x[i]=sorted_dist[i]
  print("first 3 minimum values",x[i])

for i in range(len(x)):
  for j in range(len(y)):
    if x[i] == distance[j]:
      print("index where nearest neighbours are placed:", j)
      plt.subplots()
      plt.imshow(im[:,:,:,j])
##############################
```

Output:

```
first 5 minimum values 540.4932932053829
first 5 minimum values 541.8542239385055
first 5 minimum values 549.0400713973435
first 5 minimum values 550.6659604515246
first 5 minimum values 553.060575344148
index where nearest neighbours are placed: 5
index where nearest neighbours are placed: 22
index where nearest neighbours are placed: 45
index where nearest neighbours are placed: 24
index where nearest neighbours are placed: 32
```