Machine Learning Assignment 1: K-Nearest Neighbor

Info

Attempted by Umer Abdullah Khan, Reg # 375870

Implementing K-Nearest Neighbour (KNN) algorithm to check that which actor or actress I resemble.

Tools used

Google Colab
OpenCV
NumPy

Following are the steps followed to achieve this task.

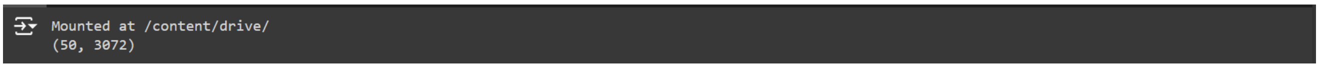
1. Linking training and test images from to Colab.

```
#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/ML/"

dataset = path + "data.mat"

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

Output:



```
Mounted at /content/drive/
(50, 3072)
```

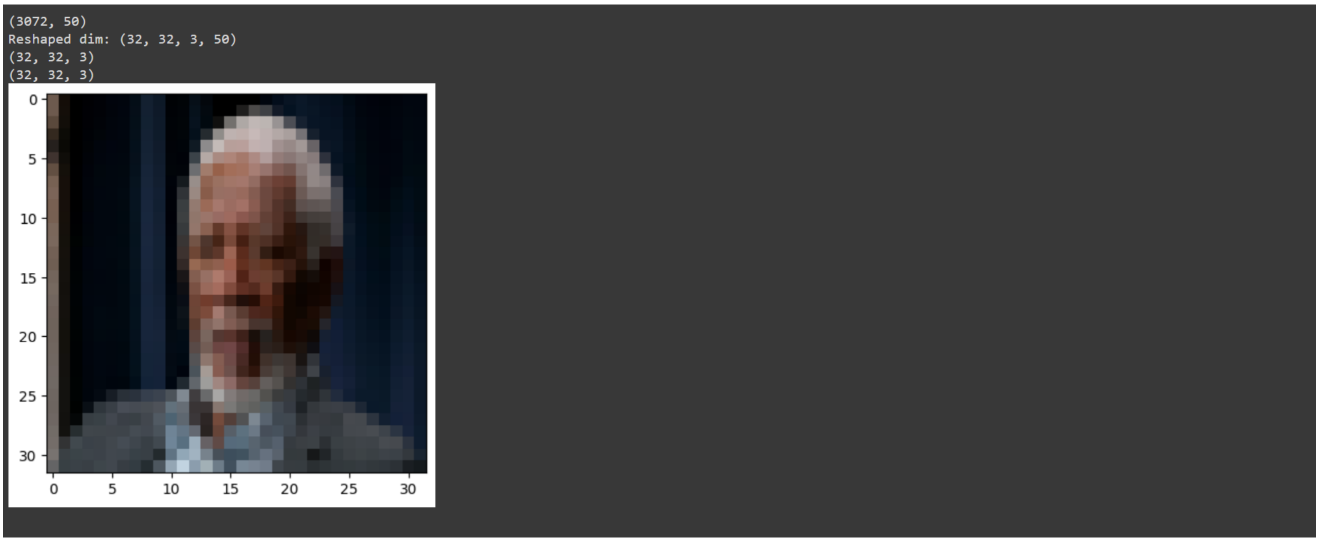
2. Ensuring data has been read.

Before doing that, we convert image dataset to 32x32x3

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

from matplotlib import pyplot as plt
plt.imshow(im[:,:,:,1])
print(im[:,:,:,0].shape)
a= im[:,:,:,1]
plt.imshow(a)
a.shape
```

Output:



```
(3072, 50)
Reshaped dim: (32, 32, 3, 50)
(32, 32, 3)
(32, 32, 3)
```

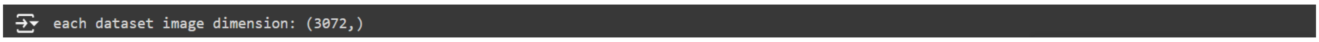
3. Convert dataset to array of 3072

```
import cv2
import numpy as np

from math import sqrt
from numpy import ndarray
from google.colab.patches import cv2_imshow

for i in range(50):
    G = im[:,:,:,i]
    G = np.reshape(G,[-1], order="F")
print("each dataset image dimension:", G.shape)
```

Output:

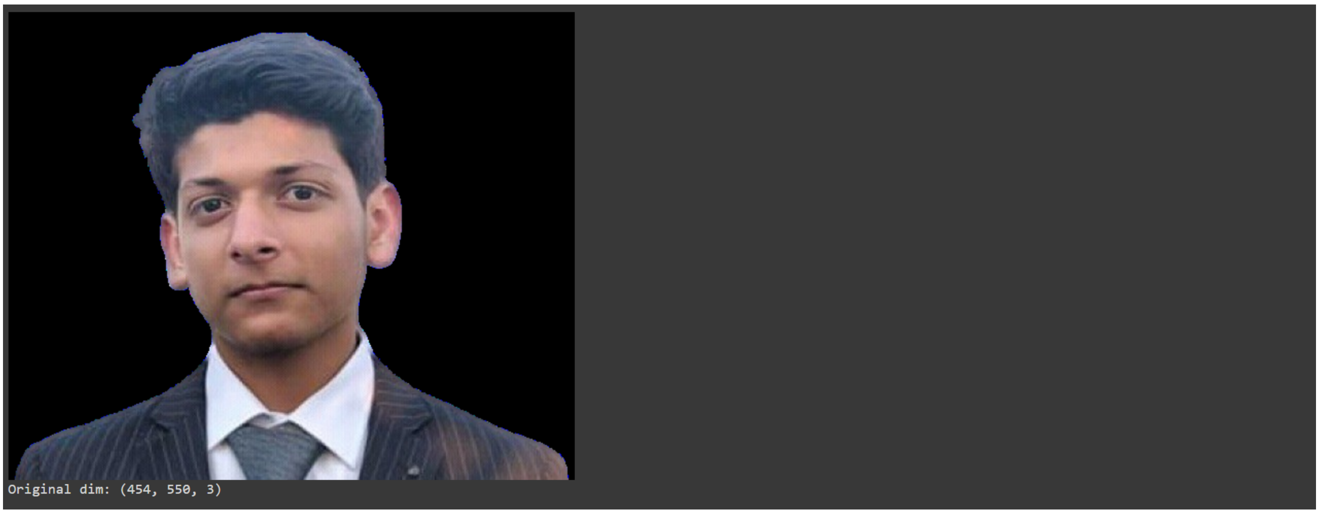


```
each dataset image dimension: (3072,)
```

4. Read and Display Test Image

```
# read image
image = cv2.imread(test_image)
# display image
cv2_imshow(image)
cv2.waitKey(0)
cv2.destroyAllWindows()
print("Original dim:", image.shape)
```

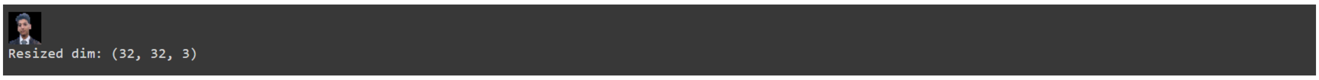
Output:



```
Original dim: (454, 550, 3)
```

5. Resize Test Image

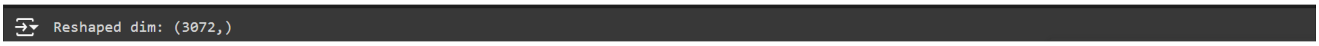
```
# resize image
resized_image=cv2.resize(image,(32,32))
# display resized image
cv2_imshow(resized_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
print("Resized dim:", resized_image.shape)
```



```
Resized dim: (32, 32, 3)
```

6. Reshaping Array Dimensions

```
reshaped_image= np.reshape(resized_image,[-1], order="F")
print("Reshaped dim:", reshaped_image.shape)
```



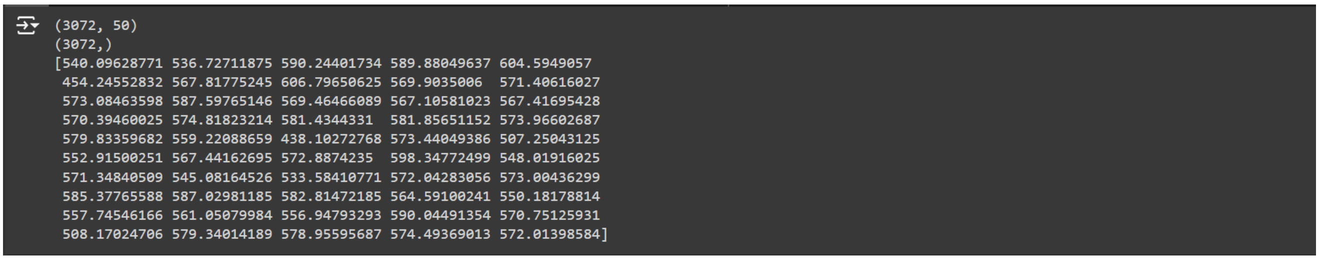
```
Reshaped dim: (3072,)
```

7. Calculate Euclidian distance between test image and dataset

```
print(images.shape)
print(reshaped_image.shape)
distances = np.zeros((50,))
for i in range(images.shape[1]):
    squared_diff = (reshaped_image - images[:, i]) ** 2
    sum_squared_diff = np.sum(squared_diff)
    distances[i] = np.sqrt(sum_squared_diff)

print(distances)
```

Output:



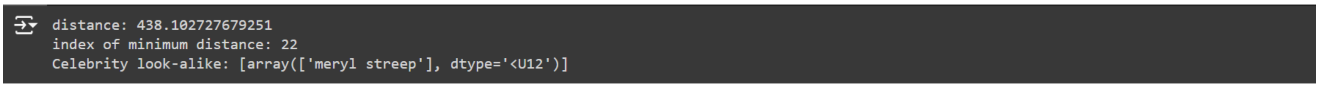
```
(3072, 50)
(3072,)
[540.09628771 536.72711875 590.24401734 589.88049637 604.5949057
 454.24552832 567.81775245 606.79650625 569.9035006  571.40616027
 573.08463598 587.59765146 569.46466089 567.10581023 567.41695428
 570.39460025 574.81823214 581.4344331  581.85651152 573.96602687
 579.83359682 559.22088659 438.10272768 573.44049386 507.25043125
 552.91500251 567.44162695 572.8874235  598.34772499 548.01916025
 571.34840509 545.08164526 533.58410771 572.04283056 573.00436299
 585.37765588 587.02981185 582.81472185 564.59100241 550.18178814
 557.74546166 561.05079984 556.94793293 590.04491354 570.75125931
 508.17024706 579.34014189 578.95595687 574.49369013 572.01398584]
```

8. Finding resembling image using 1NN

```
min_distance = min(distances)
print("distance:", min_distance)
min_index = np.argmin(distances)
print("index of minimum distance:", min_index)

name = label[min_index]
print(f'Celebrity look-alike: {name}')
```

Output:



```
distance: 438.102727679251
index of minimum distance: 22
Celebrity look-alike: [array(['meryl streep'], dtype='<U12')]
```

9. Finding resembling image using 3NN

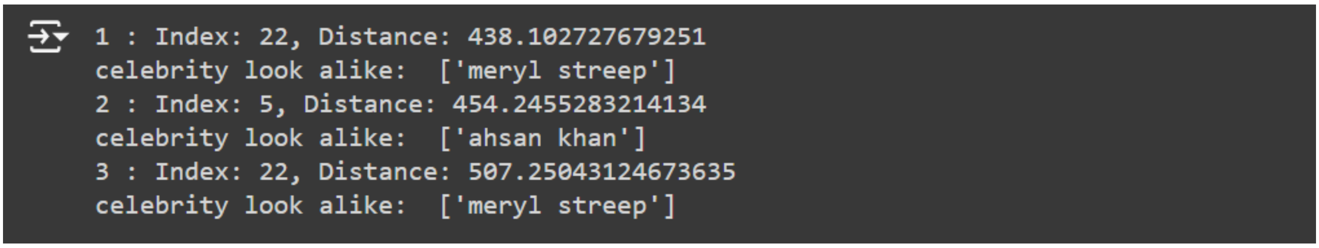
```
distances_copy = distances.copy()

min_distances = []
```

```
min_indices = []
names = []
for i in range(3):
  min_distance = min(distances_copy)
  min_index = np.argmin(distances_copy)
  min_distances.append(min_distance)
  min_indices.append(min_index)
  distances_copy = np.delete(distances_copy, min_index)
  names.append(label[min_index][0])

  print(f"{i+1} : Index: {min_indices[i]}, Distance: {min_distances[i]}")
  name = label[min_indices[i]][0]
  print("celebrity look alike: ", name)
```

Output:



```
1 : Index: 22, Distance: 438.102727679251
celebrity look alike:  ['meryl streep']
2 : Index: 5, Distance: 454.2455283214134
celebrity look alike:  ['ahsan khan']
3 : Index: 22, Distance: 507.25043124673635
celebrity look alike:  ['meryl streep']
```

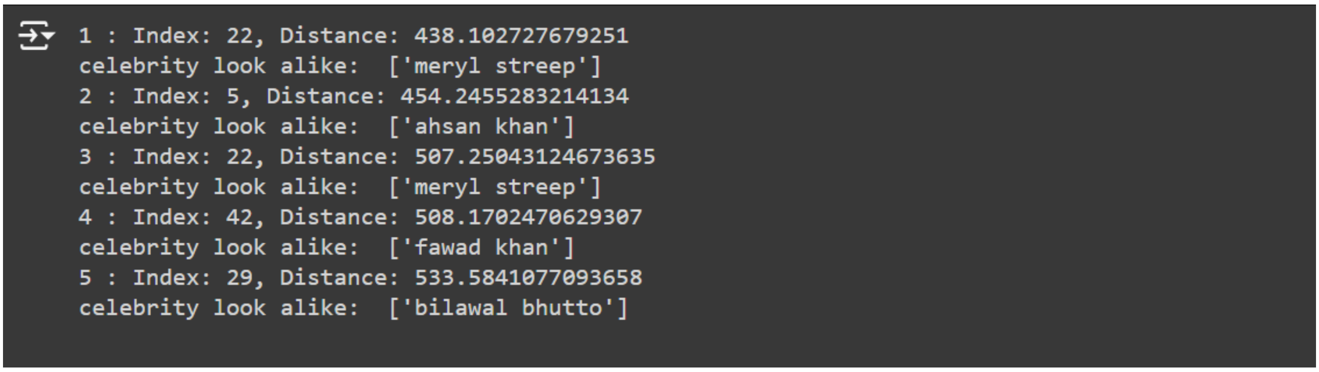
10. Finding resembling image using 5NN

```
distances_copy = distances.copy()

min_distances = []
min_indices = []
names = []
for i in range(5):
  min_distance = min(distances_copy)
  min_index = np.argmin(distances_copy)
  min_distances.append(min_distance)
  min_indices.append(min_index)
  distances_copy = np.delete(distances_copy, min_index)
  names.append(label[min_index][0])

  print(f"{i+1} : Index: {min_indices[i]}, Distance: {min_distances[i]}")
  name = label[min_indices[i]][0]
  print("celebrity look alike: ", name)
```

Output:



```
1 : Index: 22, Distance: 438.102727679251
celebrity look alike:  ['meryl streep']
2 : Index: 5, Distance: 454.2455283214134
celebrity look alike:  ['ahsan khan']
3 : Index: 22, Distance: 507.25043124673635
celebrity look alike:  ['meryl streep']
4 : Index: 42, Distance: 508.1702470629307
celebrity look alike:  ['fawad khan']
5 : Index: 29, Distance: 533.5841077093658
celebrity look alike:  ['bilawal bhutto']
```