Guided Project Report

Hashing on face dataset

Name: Shruti Verma
Course: AI and ML
(Batch 4)
Duration: 10 months

Problem Statement: Build a machine learning model using hashing on face dataset

Prerequisites

What things you need to install the software and how to install them:

Python 3.8 or higher versions This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/add-python-to-path-python-is-not- recognized-as-an-internal-or-external- command/. Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic.

Second and easier option is to download anaconda and use its anaconda prompt to run the commands. To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.8 then run below commands in command prompt/terminal to install these packages pip install -U scikit-learn pip install numpy pip install scipy if you have chosen to install anaconda then run below commands in anaconda prompt to install these packages conda install -c scikit-learn conda install -c anaconda numpy conda install -c anaconda scipy

Dataset used

The data source is Yale face dataset.There are 11 images per subject, one for each of the following facial expressions or configurations:

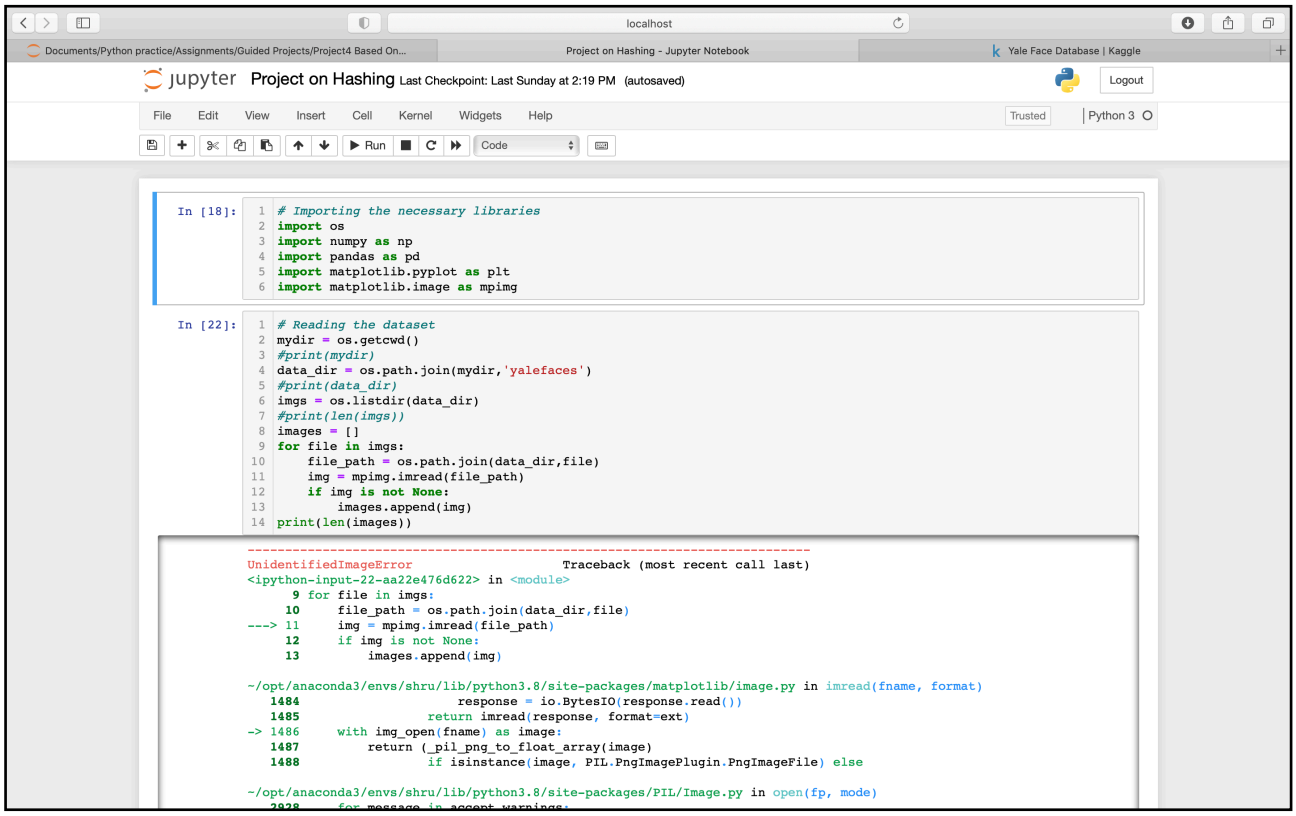
center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, sleepy, surprised, and wink.

Method used for detection

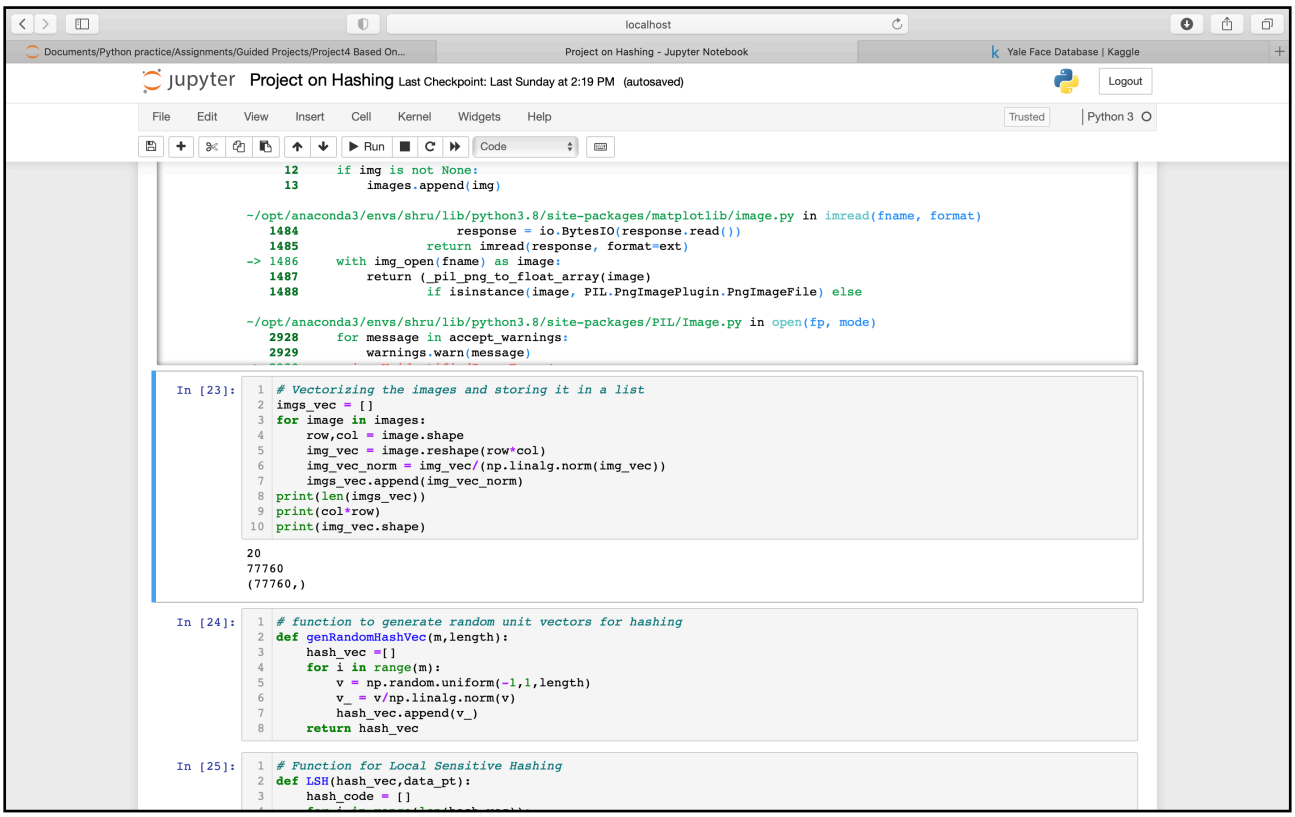
Hashing

Reading data set > LSH > plotting images with same hash code

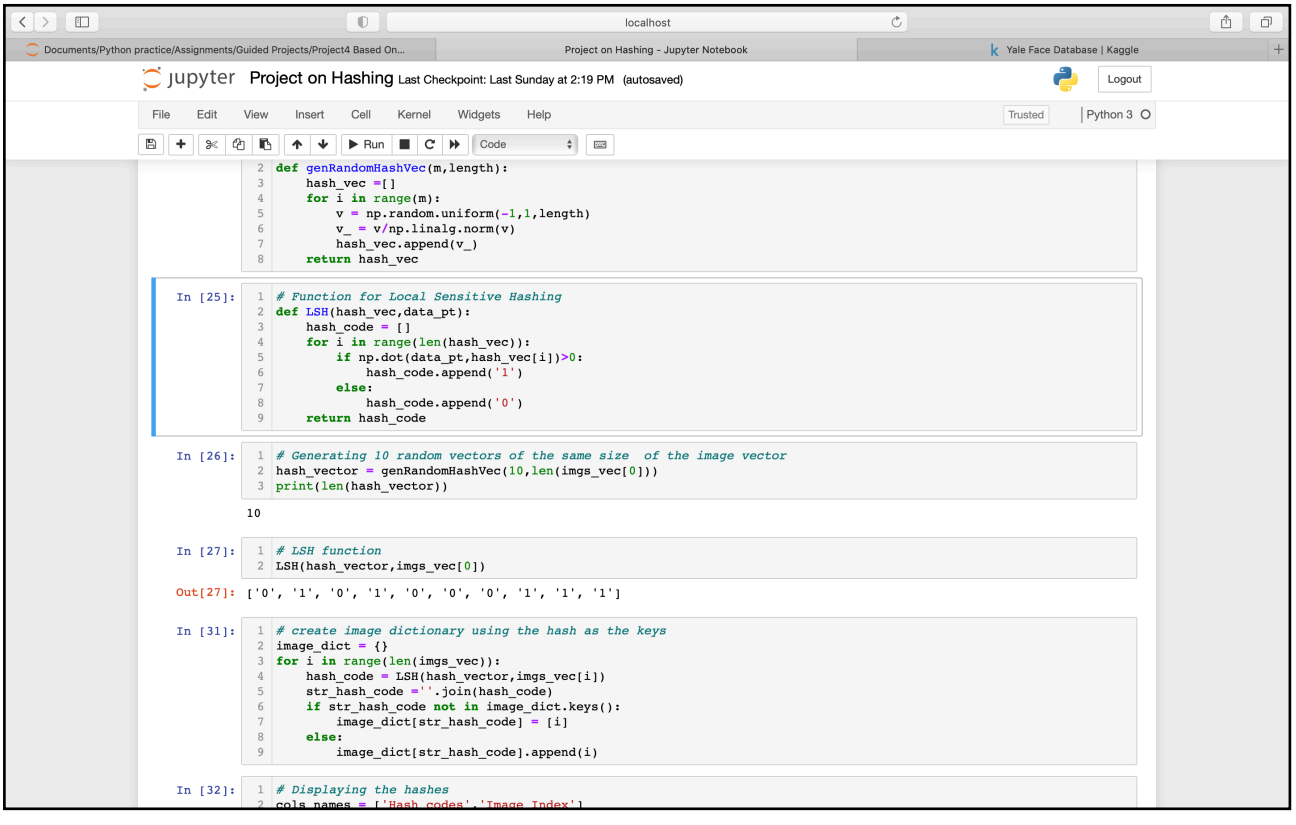
Importing the libraries and capturing images:



Vectorising the images and storing in the list



Function for Local Sensitive Hashing



Create image dictionary using the hash as the keys

localhost

Documents/Python practice/Assignments/Guided Projects/Project4 Based On... | Project on Hashing - Jupyter Notebook | Yale Face Database | Kaggle

jupyter **Project on Hashing** Last Checkpoint: Last Sunday at 2:19 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [26]:
```
# Generating 10 random vectors of the same size  of the image vector
hash_vector = genRandomHashVec(10,len(imgs_vec[0]))
print(len(hash_vector))
```

```
10
```

In [27]:
```
# LSH function
LSH(hash_vector,imgs_vec[0])
```

Out[27]:
```
['0', '1', '0', '1', '0', '0', '0', '1', '1', '1']
```

In [31]:
```
# create image dictionary using the hash as the keys
image_dict = {}
for i in range(len(imgs_vec)):
    hash_code = LSH(hash_vector,imgs_vec[i])
    str_hash_code =''.join(hash_code)
    if str_hash_code not in image_dict.keys():
        image_dict[str_hash_code] = [i]
    else:
        image_dict[str_hash_code].append(i)
```

In [32]:
```
# Displaying the hashes
cols_names = ['Hash_codes','Image_Index']
df = pd.DataFrame(image_dict.items(),columns = cols_names)
df.head(60)
```

Out[32]:

	Hash_codes	Image_Index
0	0101000111	[0]
1	1100010111	[1]
2	1101110111	[2, 4, 7]
3	1001010111	[3, 15]
4	0101010111	[5, 18]
5	0001010111	[6]
6	0100010111	[8, 13]
7	1101010111	[9, 17, 19]

Plotting the images with same hash code

localhost

Documents/Python practice/Assignments/Guided Projects/Project4 Based On... | Project on Hashing - Jupyter Notebook | Yale Face Database | Kaggle

jupyter **Project on Hashing** Last Checkpoint: Last Sunday at 2:19 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [33]:
```
# Getting the keys  and values of the Disctionary
keys = list(image_dict.keys())
values = list(image_dict.values())
```

In [34]:
```
# Plotting images with same hash code
igs = [images[i] for i in range (len(images)) if i in values[2]]
fig = plt.figure()
cols = 2
n_images =len(igs)
for n,image in zip(range(n_images),igs):
    ax = fig.add_subplot(cols,np.ceil(n_images/float(cols)),n+1)
    plt.gray()
    plt.imshow(image)
fig.set_size_inches(np.array(fig.get_size_inches())*n_images)
plt.show()
```

```
<ipython-input-34-3c80e62a9c37>:7: MatplotlibDeprecationWarning: Passing non-integers as three-element position speci
fication is deprecated since 3.3 and will be removed two minor releases later.
  ax = fig.add_subplot(cols,np.ceil(n_images/float(cols)),n+1)
```