Face and Gender Recognition project

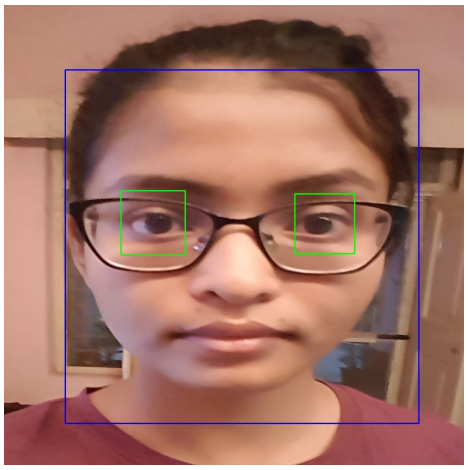
Assignment - 3

Project By
Rahil Satyanarayan Vijay(IMT2016062)
Swastik Shrivastava(IMT2016055)
Shashank Somaraju(IMT2016058)

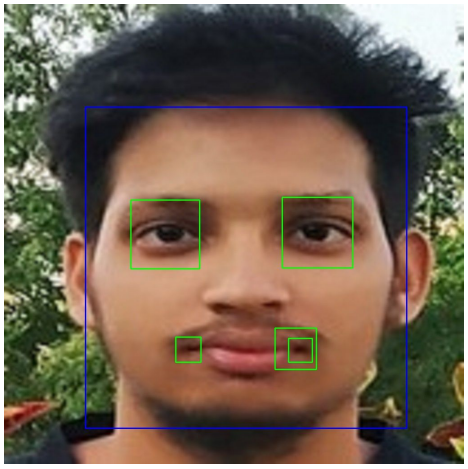
Pre-Processing:

Initially, we were using **haar-cascading** to recognise face and eyes from images, but for most of the images we were not getting satisfactory result.
Some of the results of **haar-cascade** :

Original image 1

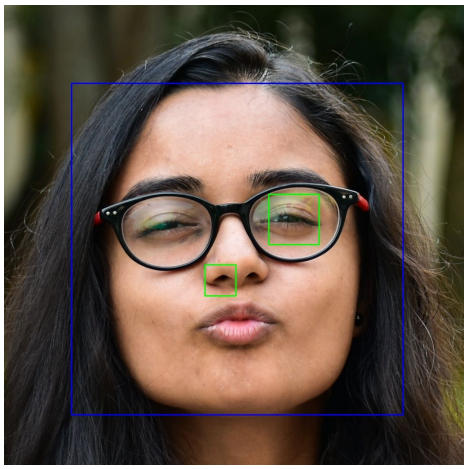


Face and eye recognition



Original image 2

Face and eye recognition



Original image 3

Face and eye recognition

Original image 4

Neither face nor eye is recognised

Due to very unreliable result of haar-cascade, we searched for other face recognition library. We came across **dlib** while searching for face recognition library. We found that the results of **dlib** were much better than haar-cascade. So we used **dlib** instead of haar-cascade.

The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face.
These annotations are part of the 68 point iBUG 300-W dataset which the dlib facial landmark predictor was trained on.

Regardless of which dataset is used, the same dlib framework can be leveraged to train a shape predictor on the input training data — this is useful if you would like to train facial landmark detectors or custom shape predictors of your own.
The indexes of the 68 coordinates can be visualized on the image below:

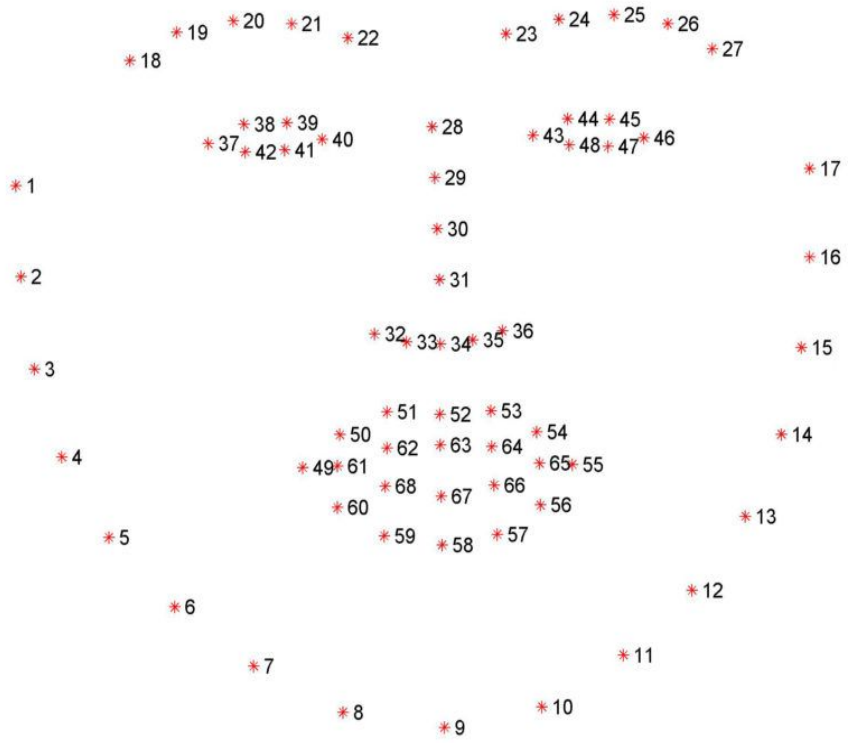
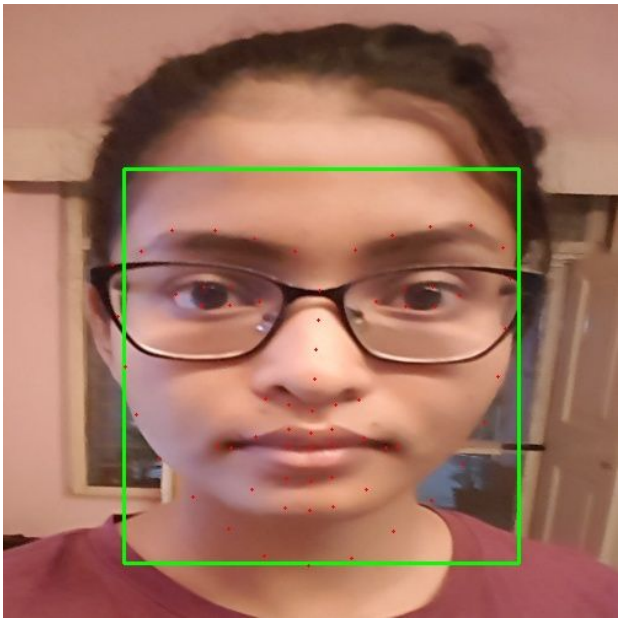
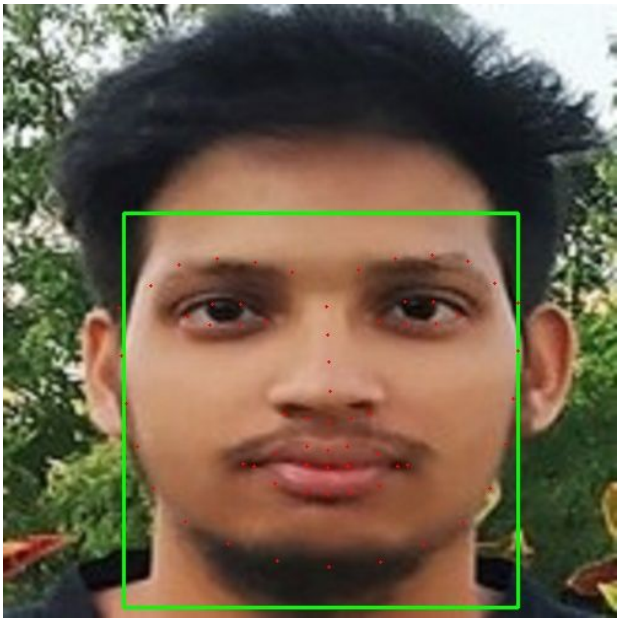


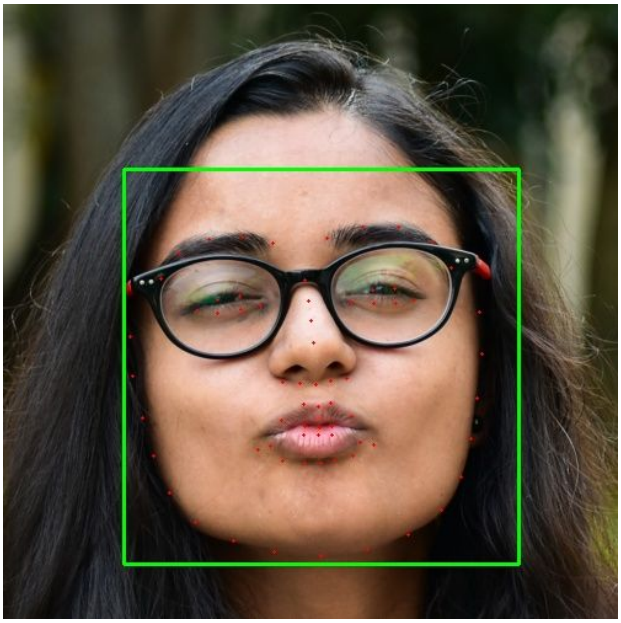
Image Coordinates

Results when dlib was used on same images:

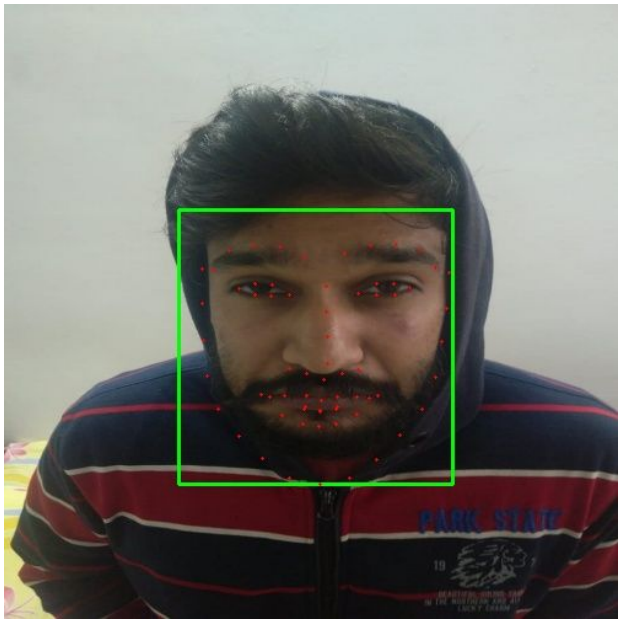




dlib on image 1



dlib on image 2



dlib on image 3

dlib on image 4

Here we can see that even the faces which haar-cascade was unable to recognise, dlib was able to recognise easily and even eye recognition is much better.

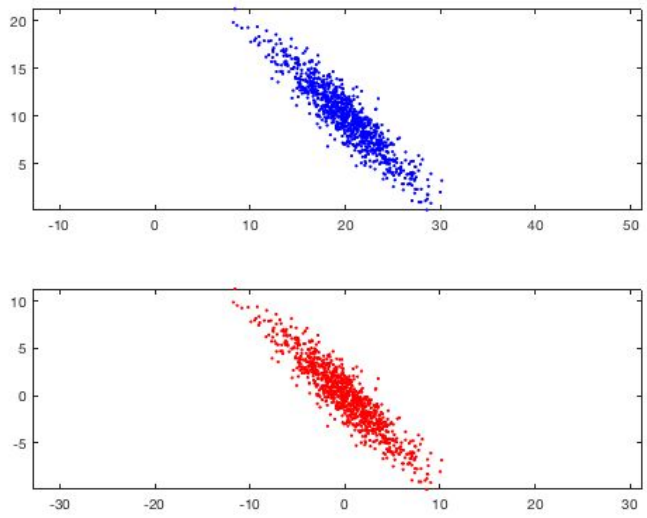
Please refer to **Image Coordinates** for coordinate points number.
To use dlib we rescaled the images to 500*500.
After using dlib on the images and getting coordinates for eyes and nose tip, we carried on with the rest of the preprocessing steps.

1. For center of the eyes we calculated the midpoint of coordinates **37** and **40** for left eye and coordinates **43** and **46** for right eye.
2. To scale the image such that the distance between eye midpoints is 128 pixels, we calculated the distance between both eyes and found out the ratio by which image should be scaled. (ratio = distance/128)
3. We rescaled the image to new dimensions. (new dimension = current dimension * ratio)
4. To get the eyes, nose and lips in a 256*256 frame we take coordinate no.30 as center of frame and crop the image.
5. At last we converted all the images to grayscale and saved in output folder.

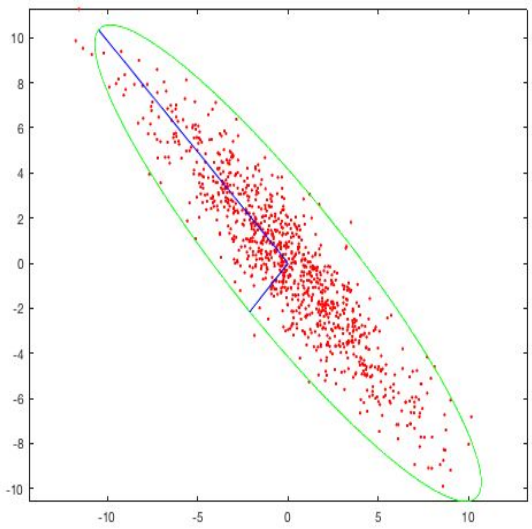
Face Recognition With PCA(Implementation of PCA and comparisons between PCA and LDA)

We're going to discuss a popular technique for face recognition called **eigenfaces**. And at the heart of eigenfaces is an unsupervised dimensionality reduction technique called **principal component analysis (PCA)**, and we will see how we can apply this general technique to our specific task of face recognition.
There are several downsides to this approach. First of all, if we have a large database of faces, then doing this comparison for each face will take a while! Imagine that we're building a face recognition system for real-time use! The larger our dataset, the slower our algorithm. But more faces will also produce better results!

We want a system that is both fast and accurate. For this, we'll use a support vector machine! We can train our network on our dataset and use it for our face recognition task.



These plots show the same data, except the bottom chart zero-centers it. Notice that our data do not have any labels associated with them because this is unsupervised learning! In our simple case, dimensionality reduction will reduce these data from a 2D plane to a 1D line. If we had 3D data, we could reduce it down to a 2D plane or even a 1D line.
All dimensionality reduction techniques aim to find some hyperplane, a higher-dimensional line, to project the points onto. We can imagine a projection as taking a flashlight perpendicular to the hyperplane we're project onto and plotting where the shadows fall on that hyperplane. For example, in our above data, if we wanted to project our points onto the x-axis, then we pretend each point is a ball and our flashlight would point directly down or up (perpendicular to the x-axis) and the shadows of the points would fall on the x-axis



The idea behind PCA is that we want to select the hyperplane such that when all the points are projected onto it, they are maximally spread out. In other words, we want the *axis of maximal variance!*Let's consider our example plot above. A potential axis is the x-axis or y-axis, but, in both cases, that's not the best axis. However, if we pick a line that cuts through our data diagonally, that is the axis where the data would be most spread.After this we applied the SVM(Linear) classifier giving us the results as follows:

```
(648, 196608)
(161, 196608)
0.5900621118012422
Top  10  accuracy:  0.8633540372670807
Top  5  accuracy:  0.7701863354037267
Top  3  accuracy:  0.7018633540372671
Top  1  accuracy:  0.577639751552795
```

PCA Results

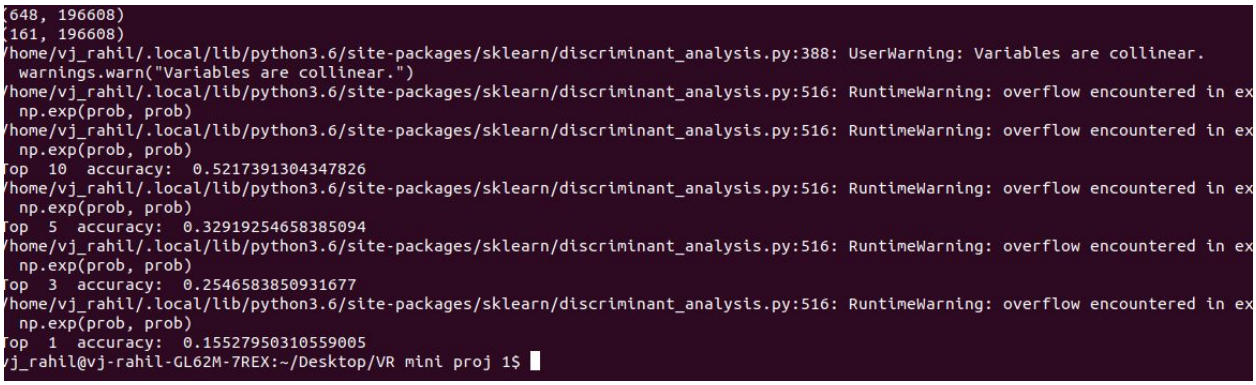
First Line is X_train.shape
Second Line is X_test.shape
Dimension reduction to 63
Score = 59.006%
Top -10 = 86.33
Top -5 = 77.01%

Top -3 = 70.1%
Top -1 = 57.7%

PCA vs LDA

Both Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) are linear transformation techniques that are commonly used for dimensionality reduction. PCA can be described as an “unsupervised” algorithm, since it “ignores” class labels and its goal is to find the directions (the so-called principal components) that maximize the variance in a dataset. In contrast to PCA, LDA is “supervised” and computes the directions (“linear discriminants”) that will represent the axes that that maximize the separation between multiple classes.

LDA Results



I got some warning in the code but it’s given the correct accuracy
Line 1 is X_train.shape
Line 2 is X_test.shape
Top -10 = 52.1%
Top -5 = 32.9%
Top -3 = 25.4%
Top -1 = 15.5%

Question 4:

Using Docker image (open source image), we ran the docker container and installed all the binaries in it. Then we linked the running Docker container to our raw data folder. After getting pre-processed data by running the script in the docker container, we got 2 CSV files as features namely - “label.csv” and “reps.csv”. Then we wrote a script in python to predict the score using these CSV files.

```
3 q1.py
/home/shashank/anaconda3/envs/vr/lib/python3.5/site-packages/sklearn/model_selec
tion/_split.py:626: Warning: The least populated class in y has only 3 members,
which is too few. The minimum number of members in any class cannot be less than
 n_splits=4.
  % (min_groups, self.n_splits)), Warning)
/home/shashank/anaconda3/envs/vr/lib/python3.5/site-packages/sklearn/model_selec
tion/_search.py:841: DeprecationWarning: The default of the `iid` parameter will
 change from True to False in version 0.22 and will be removed in 0.24. This wil
l change numeric results when test-set sizes are unequal.
  DeprecationWarning)
0.9140625
```

This gave us an accuracy of **0.9140625** We used OpenFace again for predicting gender, we got an accuracy of **0.93451**. OpenFace pipeline uses 500,000 images which are passed through the neural net. These images are from two public datasets: CASIA-WebFace, which is comprised of 10,575 individuals for a total of 494,414 images and FaceScrub, which is made of 530 individuals with a total of 106,863 images. OpenFace uses Google’s FaceNet architecture for feature extraction and uses a triplet loss function to test how accurate the neural net classifies a face. It does this by training on three different images where one is a known face image called the anchor image, then another image of that same person has positive embeddings, while the last one is an image of a different person, which has negative embeddings. With OpenFace we can still use dlib as we did for face detection. We can combine it with HOG and SVM.

Question 5:

Identifying demographic attributes of humans such as age, gender and ethnicity using computer vision has been given increased attention in recent years. Such attributes can play an important role in many applications such as human-computer interaction, surveillance, content-based indexing and searching, biometrics, demographic studies and targeted advertising. For example, in face recognition systems, the time for searching the face database can be reduced and separate face recognizers can be trained for each gender to improve accuracy.

In general, a pattern recognition problem such as gender recognition, when tackle with a supervised learning technique, can be broken down into several steps which are object detection, preprocessing, feature extraction and classification. In detection, the human subject or face region is detected and cropped from the image.
This is followed by some preprocessing, for example geometric alignment, histogram equalization or resizing. In feature extraction, representative descriptors of the image are

found, after which selection of the most discriminative features may be made or dimension reduction is applied.Lastly, the classifier is trained and validated using a dataset. As the subject is to be
classified as either male or female, a binary classifier is used, for example, support vector machine (SVM), Adaboost, neural networks and Bayesian classifier.

GENDER RECOGNITION BY FACE:

The face region, which may include external features such as the hair and neck region, is used to make gender identification. The image of a person's face exhibits many variations which may affect the ability of a computer vision system to recognize the gender, which can be categorized as being caused by the image capture process other human. Factors due to the former are the head pose or camera view , lighting and image quality. Head pose refers to the head orientation relative to the view of the image capturing device, as described by the pitch, roll and yaw angles.

Feature extraction

We broadly categorize feature extraction methods for face gender classification into geometric-based and appearance-based methods . The former is based on distance measurements of fiducial points, which are important points that mark features of the face, such as the nose, mouth, and eyes. Psychophysical studies using human subjects established the importance of these distances in discriminating gender. While the geometric relationships are maintained, other useful information may be discarded and the points need to be accurately extracted .Appearance-based methods are based on some operation or transformation performed on the image pixels, which can be done globally (holistic) or locally (patches). The geometric relationships are naturally maintained , which is advantageous when gender discriminative features are not exactly known. But they are sensitive to variations in appearance (view, illumination, etc.) and the large number of features . Pixel intensity values can be directly input to train a classifier such as neural network or support vector machine (SVM).

GENDER RECOGNITION BY GAIT

Gait is defined to be the coordinated, cyclic combination of movements that result in human locomotion , which includes walking, running, jogging and climbing stairs. In computer vision research, the gait of a walking person is often used. Exploiting gait information is helpful in some situations such as when the face is not visible. In a video

sequence of a person walking, the gait cycle can be referred to as the time interval between two consecutive left/right mid-stances [45]. Many factors affect the gait of a person, such as load , footwear, walking surface, injury, mood, age
and change with time. Video-based analysis of gait would also need to contend with clothing, camera view , walking speed and background clutter.

Feature extraction

Early work on gait analysis used point lights attached to the body's joints. Based on the motion of the point lights during walking, identity and gender of a person could be identified . Human gait representation can be divided into model-based or appearance-based (model-free). One way to do is by obtained 2-D stick figures from the body contour, of which a sequence from one gait cycle composed a gait signature. Such model-based approaches rely on accurate estimation of joints and require high-quality gait sequences where the body parts need to be tracked in each frame, thus incurring higher computational costs. Moreover, they ignore body width information . However, they are view and scale invariant .In many appearance-based methods, the silhouette of the human is obtained, for example using background subtraction.

Gender Recognition by Body

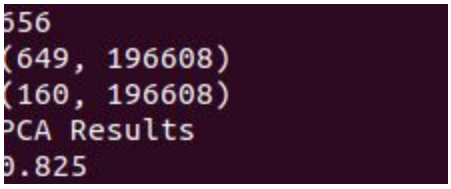
Here, we refer to the use of the static human body (either partially or as a whole) in an image which, like gait, would be useful in situations where using the face is not possible or preferred. However it is challenging in several aspects. To infer gender, humans use not only body shape and hairstyle, but additional cues such as type of
clothes and accessories , which may be the similar among different genders. The classifier should also be robust to variation in pose, articulation and occlusion of the person and deal with varying illumination and background clutter.

Feature extraction

The first attempt to recognize gender from full body images partitioned the centered and height-normalized human image into patches corresponding to some parts of the body . Each part was represented using Histogram of Oriented Gradients (HOG) feature, which was previously developed for human detection in images . HOG features are able to capture local shape information from the gradient structure with easily controllable degree of invariance to translations or rotations .

A Possible Solution for gender recognition:

The first step is to do preprocessing. We are already doing preprocessing for the given dataset for the face recognition.Now for the recognition algorithm there are some methods by which we can implement the gender recognition.Gender recognition using OpenCV's fisherfaces implementation is quite popular. But implementing using CNN will be easy and effective for our case . For our dataset we need to label each photo as male or female to train the CNN. For that we can use dictionaries and match each name with male or female . Now we do cross validation and train the data and get the accuracies and compare for the best model.



Gender Recognition Result

Line 2 is X_train.shape
Line 3 is X_test.shape
Accuracy = 82.5%

How to run the code

1)First run the q.py file it will create output from AVR_data folder
2)Run pca.py to get the accuracies prediction for PCA and LDA model
3)For gender recognition run the gender.py to get accuracy prediction
4)When you run Opensource.py with the label.csv and rvps.csv it will give prediction for the model in the docker.

Reference

1)https://pythonmachinelearning.pro/face-recognition-with-eigenfaces/
2)https://sebastianraschka.com/Articles/2014_python_lda.html
3)https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/
4)https://cmusatyalab.github.io/openface/
5)https://datascience.stackexchange.com/questions/18804/sklearn-select-n-best-using-classifier