Alexandria University- Faculty of Engineering

Computer and Systems Engineering Department

Pattern Recognition

Lab 1

Face Recognition

Name	ID	Email
Ahmed Gamal Mahmoud	18010083	ahmed.gamal5551.ag@gmail.com
Marwan Mohamed Saad	18011736	marwangabal99@gmail.com
Shehab Mohamed Saad	18010857	shehab.mohamed2104@gmail.com

colab link

Downloading dataset and understanding the format:

- We uploaded the dataset on google drive mount the dataset folder to our colab
- Every image in the dataset is in format of pgm with dimensions 92x112 so reading the image in grayscale is loaded in 2d array
- This images needed a reshaping to be dealt with

Generating data matrix & label vector:

- We loop over the images to reshape the images into 1d array so the new dimension of the image became 10304
- The data matrix D is a matrix of all images after reshaping
- We generate the label vector during reading of images to keep track of every person id (folder name)

Splitting dataset into training & testing:

- The images in D are split into 2 halves one for training and one for testing
- That is done through looping over D and every row id is to be checked if is even then we append to testing list else append to training list

Classification using PCA:

1. Functions :
    a. pca_values_vectors :
        i. Takes the X_train data as input and calculates it's mean vector
        ii. Centralizing the data by it's mean vector
        iii. Calculates the covariance matrix and get the eigen values and eigen vectors corresponding to the eigen values and returns them
    b. pca_accuracies_bestTrainTest
        i. Take the eigenvalues , eigenvectors , X_train , X_test , Y_train and Y_test

ii. Going to a loop to get the **r** which indicates the number of largest eigenvalues and it's corresponding eigenvectors to meet the required alpha

iii. Project the training set and testing set after getting the suitable **r**

iv. Using First nearest neighbor (k = 1) to classify the projected data and get the accuracy

v. Repeat from step **(ii)** for every alpha and get it's accuracy

vi. Returns the list of accuracies / best accuracy among the 4 alphas / best Y_pred list of the best alpha / best projected train set / best projected testing set

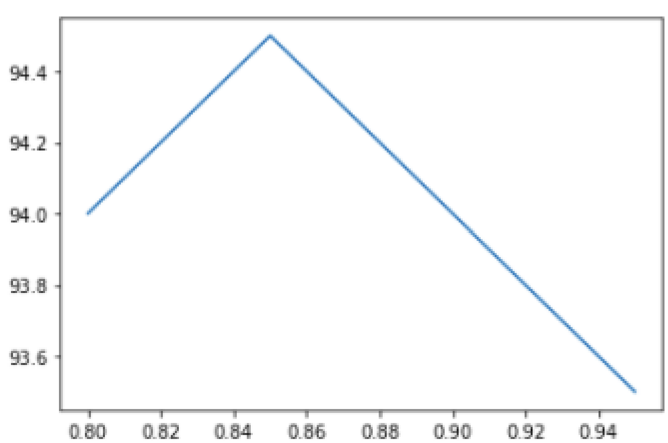
c. PCA

i. Just calling the above 2 functions to sum up everything

2. Accuracy summary for every alpha :

Alpha	0.8	0.85	0.9	0.95
Accuracy	94 %	94.5 %	94 %	93.5 %

3. Relation between accuracy and alpha :



There is no correlation between alpha and accuracy . sometimes the accuracy increase while alpha increase but then the accuracy decrease while the alpha increase. Therefore there is no relation.

Classification using LDA:

1. Functions:

   A. eig_val_vec_LDA

   I. Takes X_train, n (number of classes), nk (array of instances size of each class), c (dimensions of pic 10304).
   II. Computes mean for each class
   III. Computes overall mean
   IV. Computes Sb
   V. Computes S
   VI. Computes S inverse
   VII. Computes S^{-1} * Sb
   VIII. Computes eigen values and vectors

   B. get_projected_data_LDA

   I. Takes start_index (the index of first vector in eigen vectors will be taken), eigenvectors, X_train, X_test.
   II. Computes Projection matrix
   III. Get the projected training set
   IV. Get the projected testing set

   C. get_accuracy_LDA

   I. Takes new_train (projected training set), new_test (projected testing set), Y_train, Y_test.
   II. Classify using KNN with neighbors = 1
   III. Return Accuracy and Y_pred list.

2. Accuracy : 95.5% **which is greater than PCA**

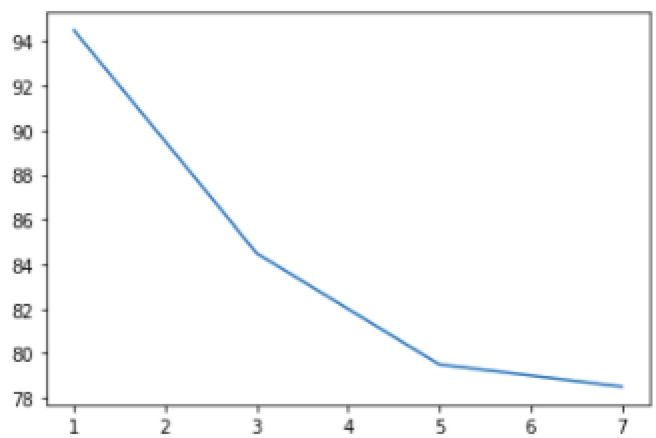
Classifier Tuning

1. PCA

    a. Trying KNN with k = 1 , 3 , 5 , 7

    b.

K	1	3	5	7
Accuracy	94.5 %	84.5 %	79.5 %	78.5 %

    c. 

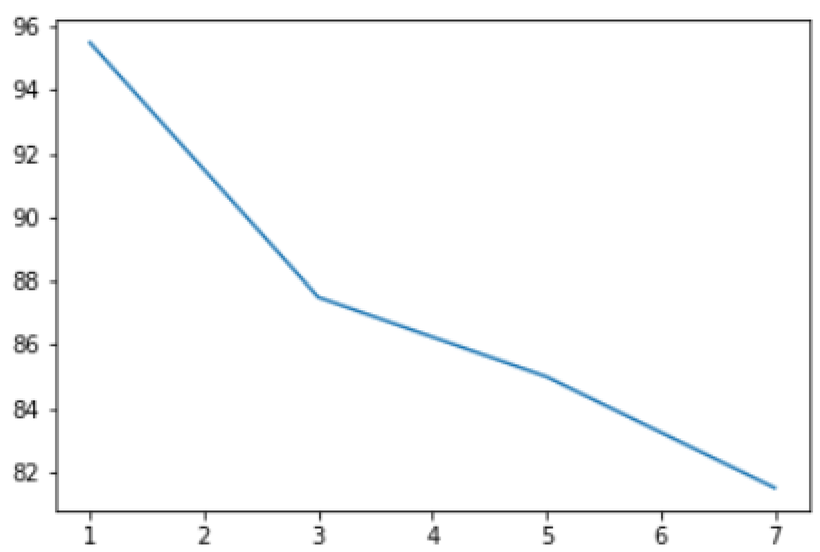
    d. From the plotting : we deduce the best accuracy is gained when K = 1 and it keep decreasing while K increases

2. LDA

    a. Trying KNN with k = 1 , 3 , 5 , 7

b.

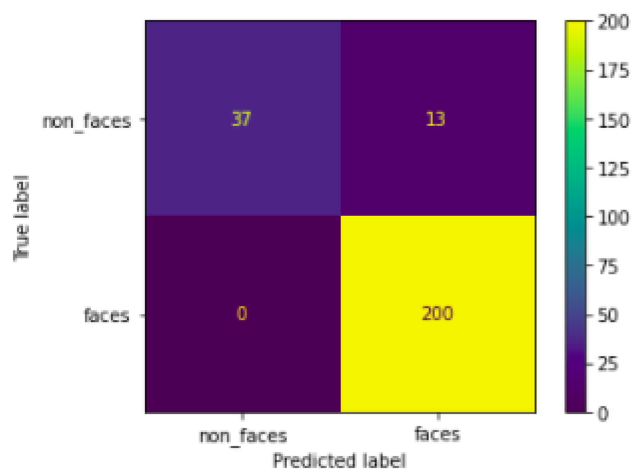
K	1	3	5	7
Accuracy	95.5 %	87.5 %	85.0 %	81.5 %

c. 

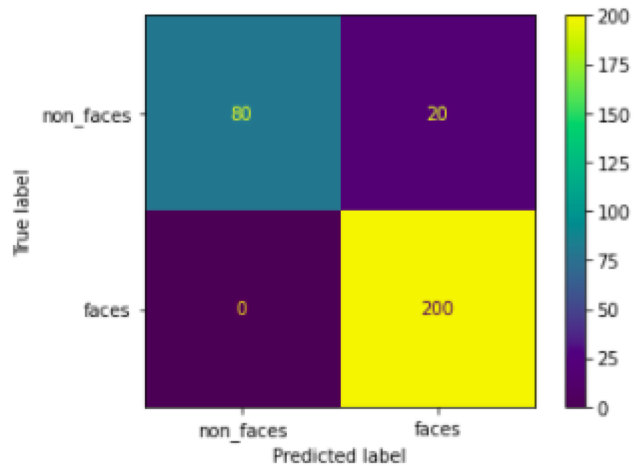
d. From the plotting : we deduce the best accuracy is gained when K = 1 and it keep decreasing while K increases

Compare non faces vs faces

1. PCA
    a. Success and failure cases while fixing faces at 400 images
        i. 100 image of non faces

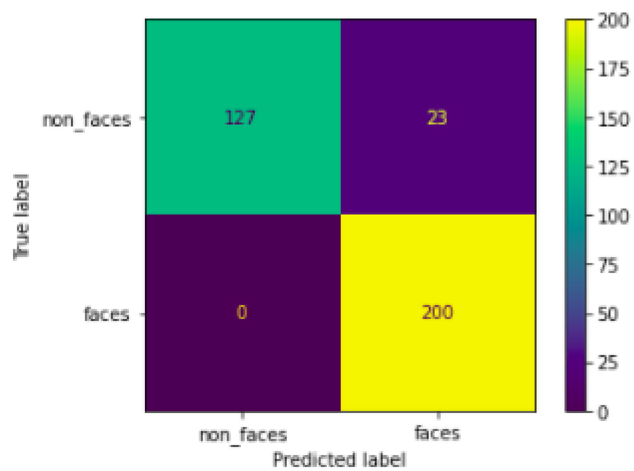
1. 

ii. 200 image of non faces



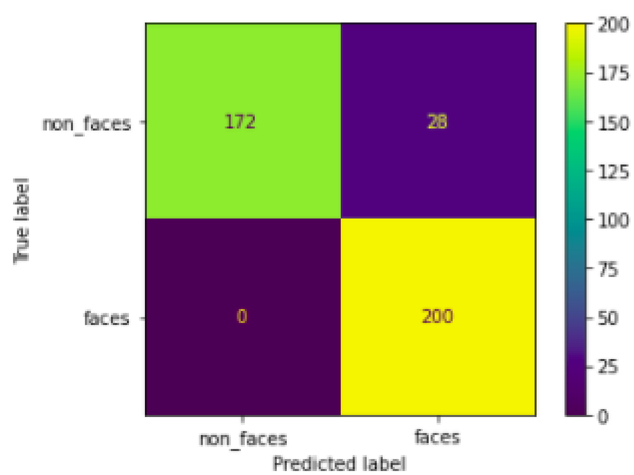
1.

iii. 300 image of non faces



1.

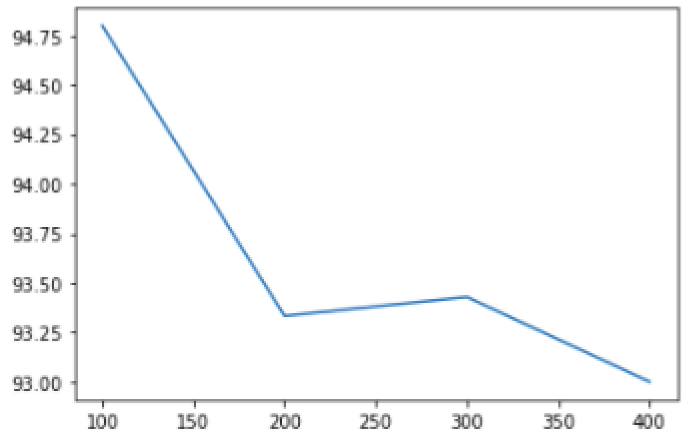
iv. 400 image of non faces



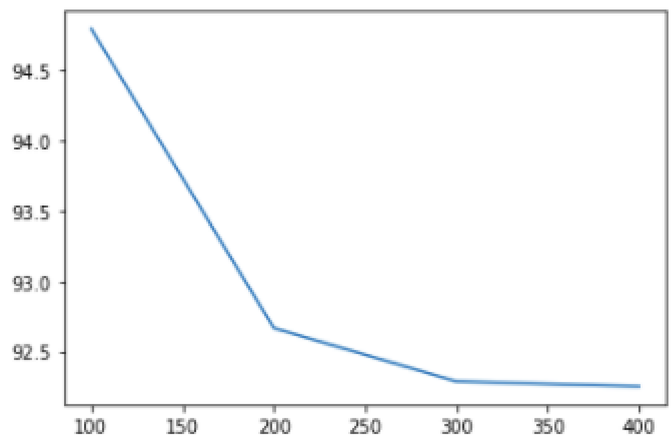
1.

b. Accuracy while fixing faces at 400 image (Accuracy At Y -axis and number of images of non faces at X-axis)

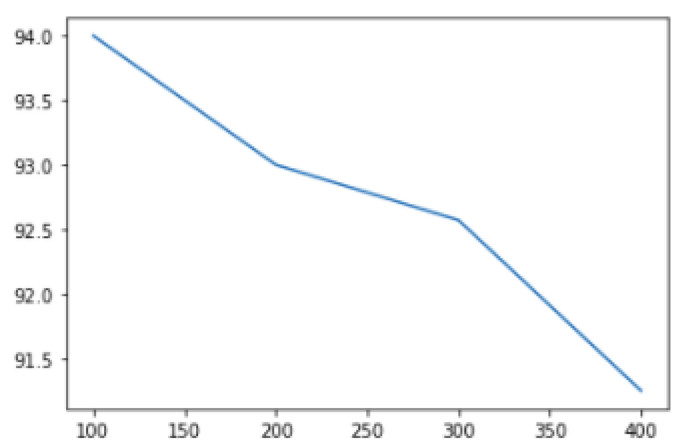
i. At alpha = 0.8

1. 

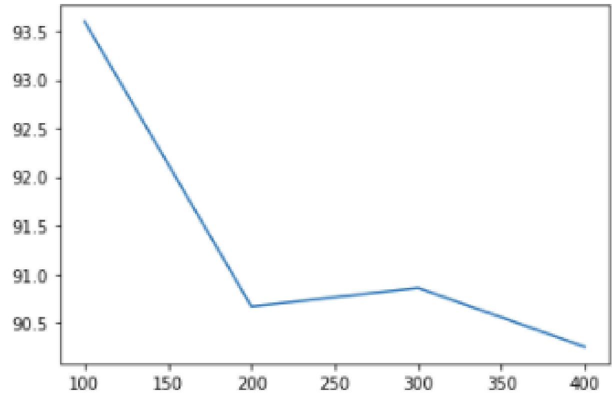
ii. At alpha = 0.85

1. 

iii. At alpha = 0.9



iv. At alpha = 0.95

1. 

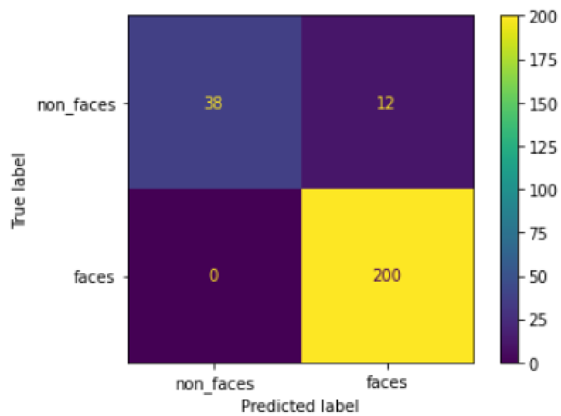
2. LDA

Seed of shuffling data = 4

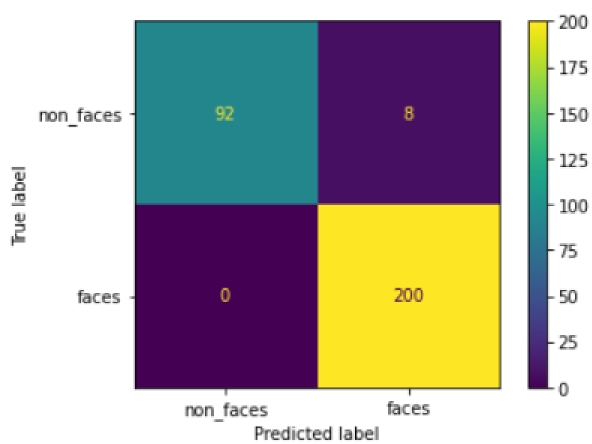
a. When we tune eigen vectors using loop from 1 to 50:

i. <u>Success and failure cases while fixing faces at 400 images</u>

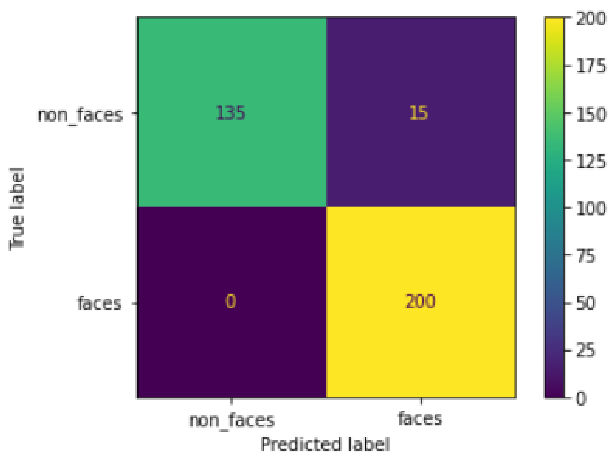
1. Non faces 100



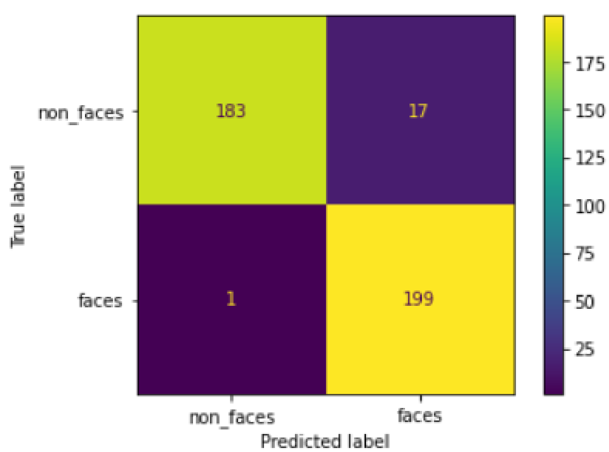
2. Non faces 20



3. Non faces 300

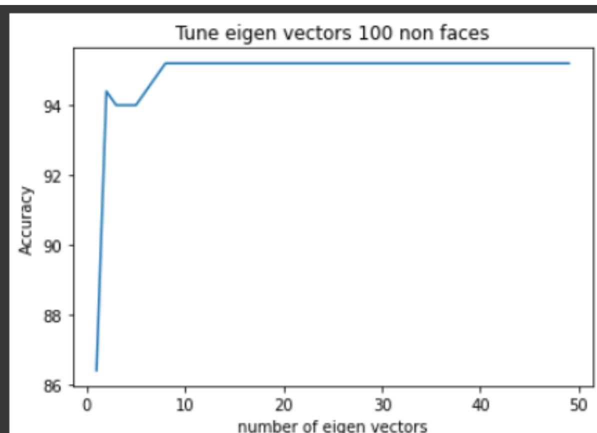


4. Non faces 400



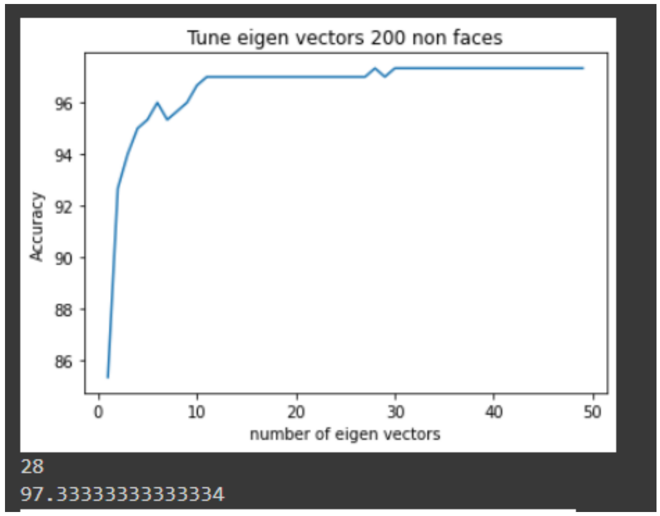
ii. **How many dominant eigenvectors will you use for the LDA solution?**

1. Non faces 100

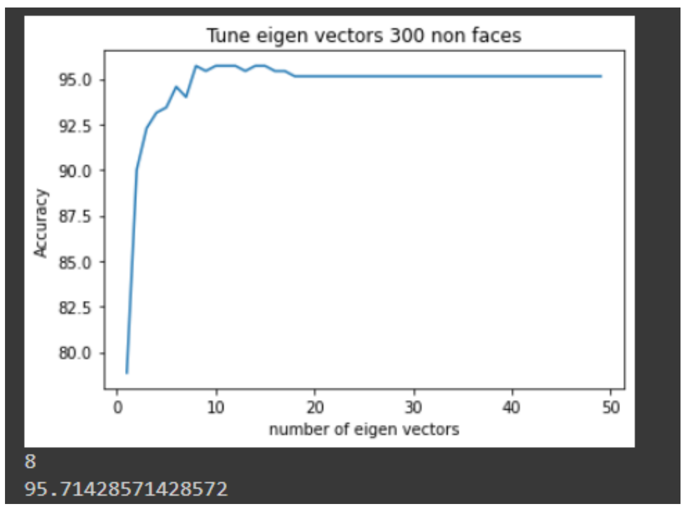



```
8
95.19999999999999
```

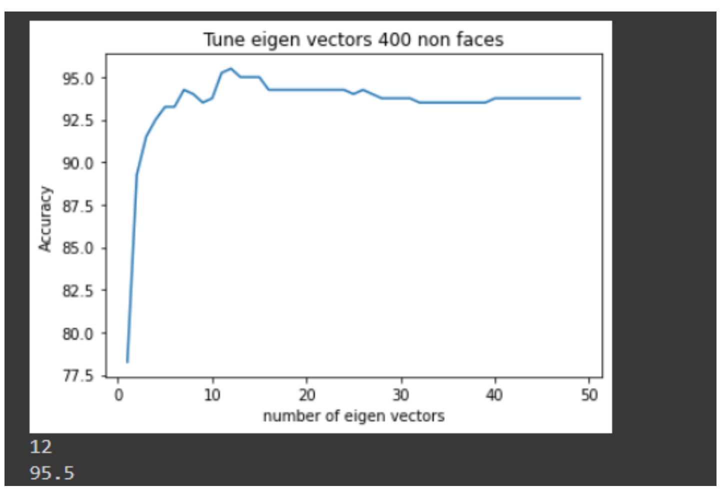
2. Non faces 200



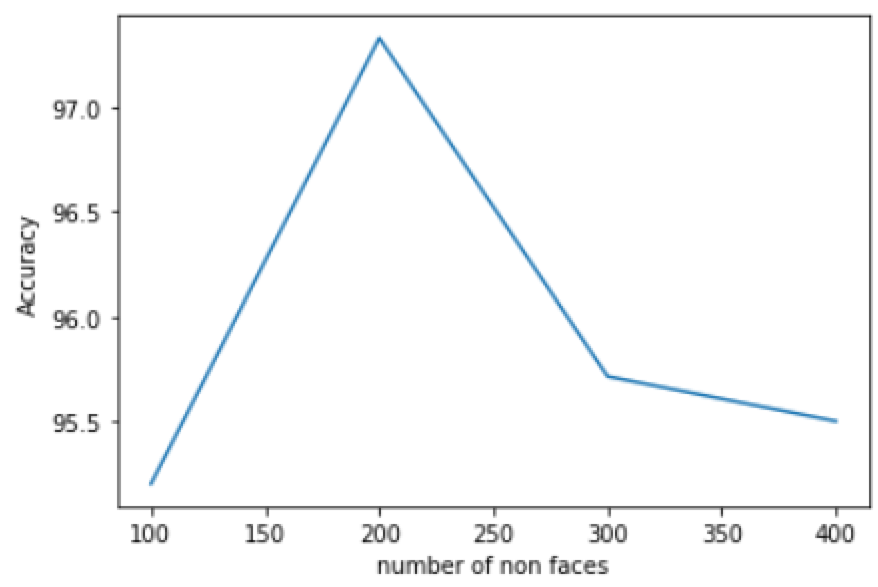
3. Non faces 300



4. Non faces 400



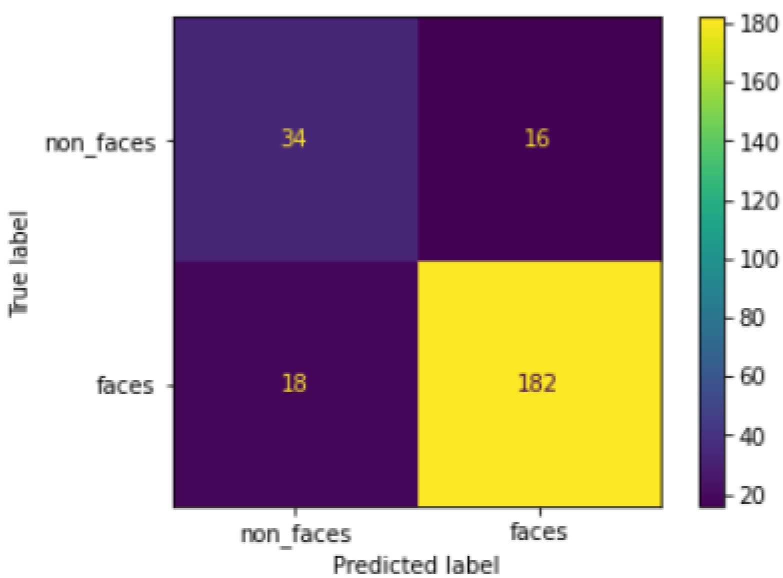
iii. Accuracy while fixing faces at 400 image (Accuracy At Y -axis and number of images of non faces at X-axis)

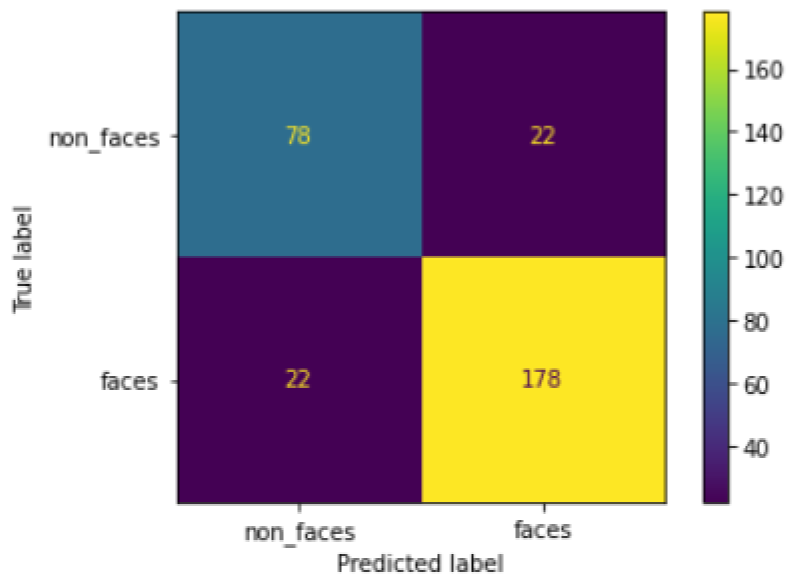


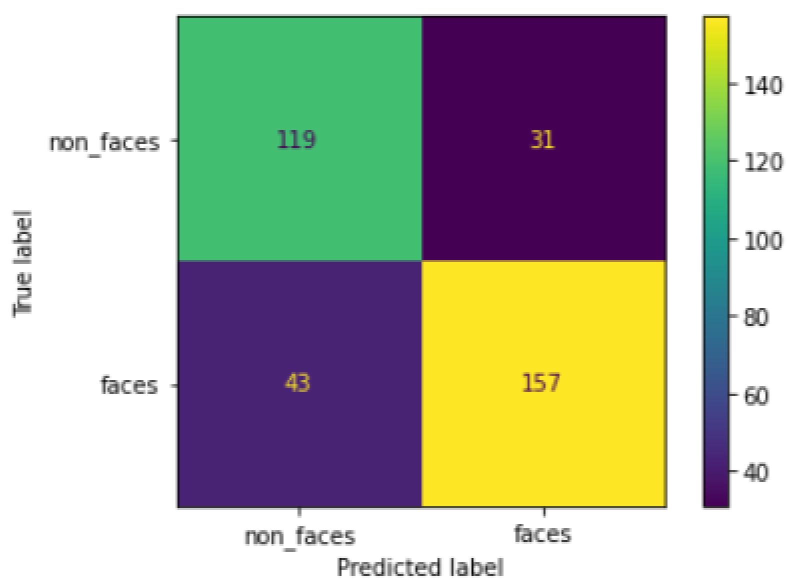
b. When taking eigen vectors (# classes -1 which equals 1)

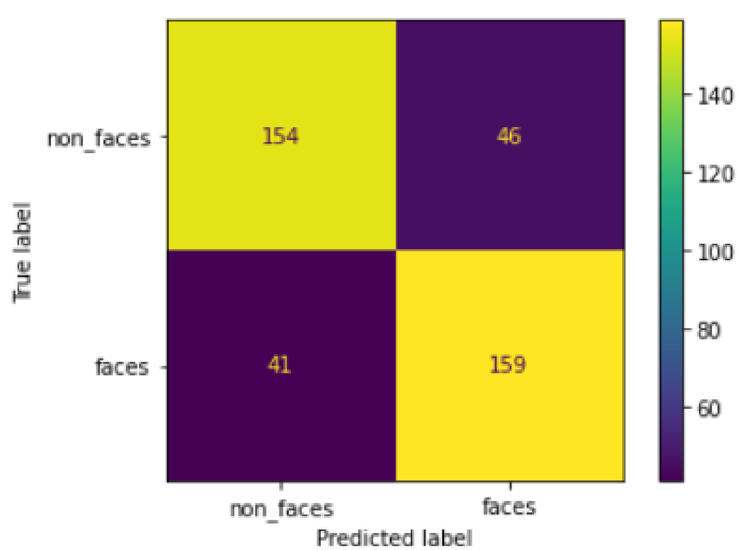
i. Success and failure cases while fixing faces at 400 images

1. Non facing equals 100, 200, 300, 400 in order

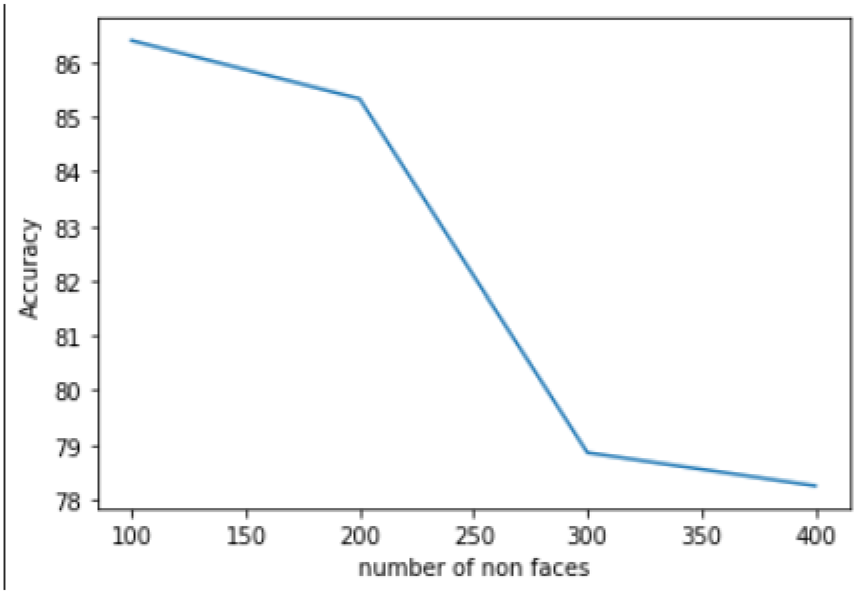



non_faces
faces
True label
78
22
22
178
non_faces
faces
Predicted label
160
140
120
100
80
60
40


non_faces
faces
True label
119
31
43
157
non_faces
faces
Predicted label
140
120
100
80
60
40


non_faces
faces
True label
154
46
41
159
non_faces
faces
Predicted label
140
120
100
80
60

ii. Accuracy while fixing faces at 400 image (Accuracy At Y -axis and number of images of non faces at X-axis)

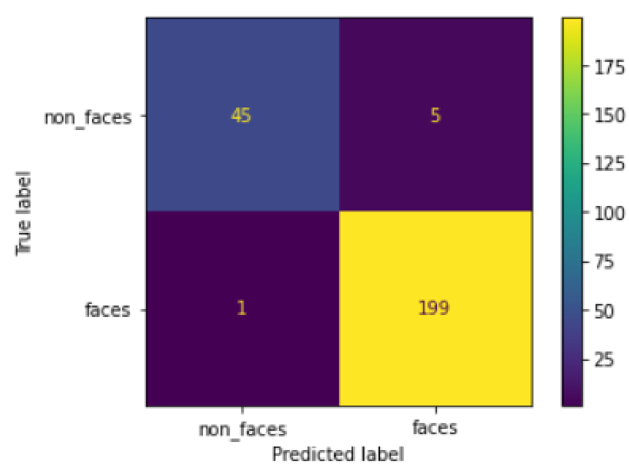


Seed of shuffling data = 7

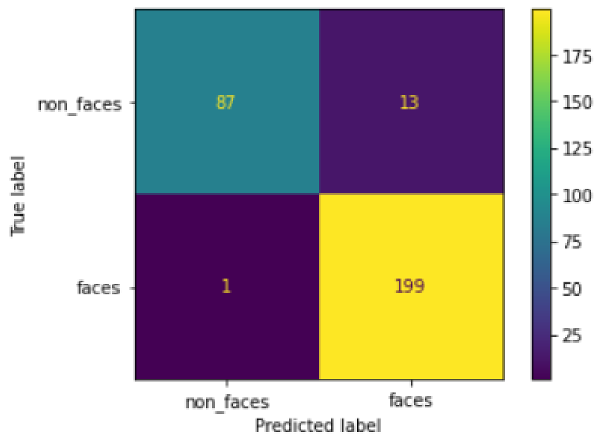
A. When we tune eigen vectors using loop from 1 to 50:

1. Success and failure cases while fixing faces at 400 images

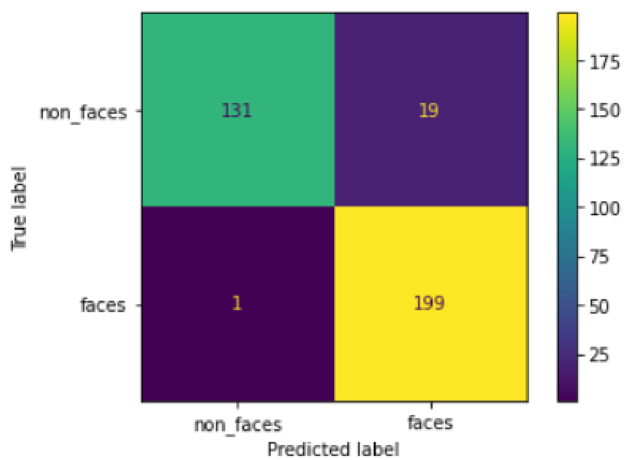
a. Non faces 100



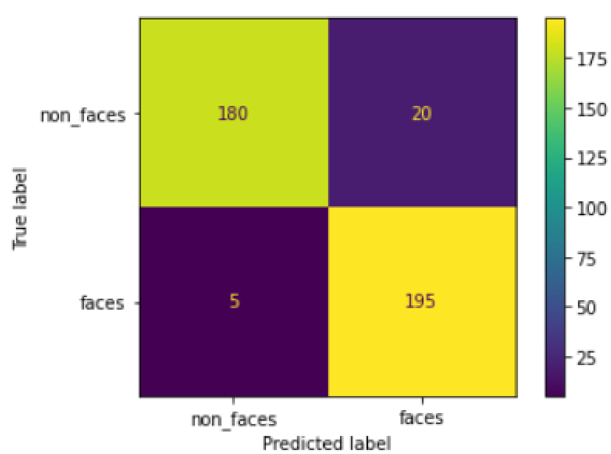
b. Non faces 200



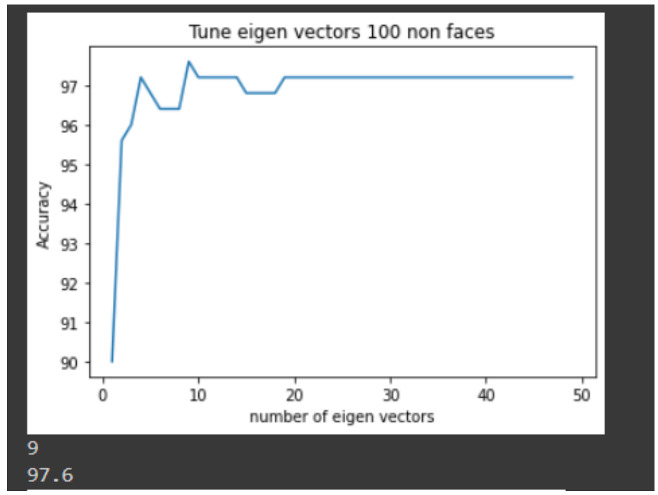
c. Non faces 300



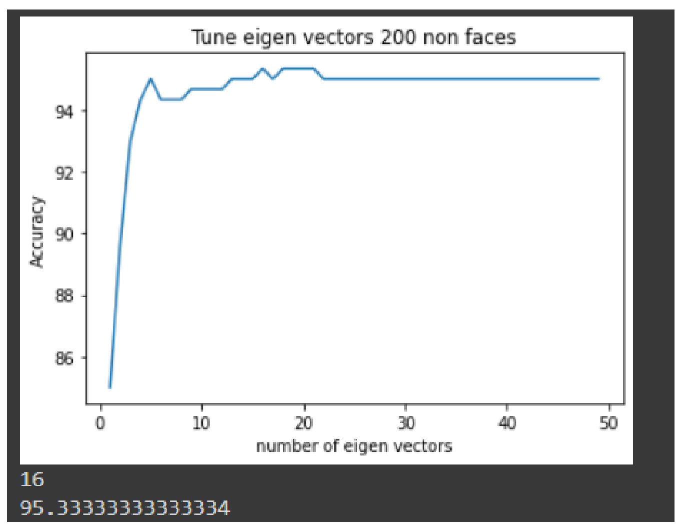
d. Non faces 400



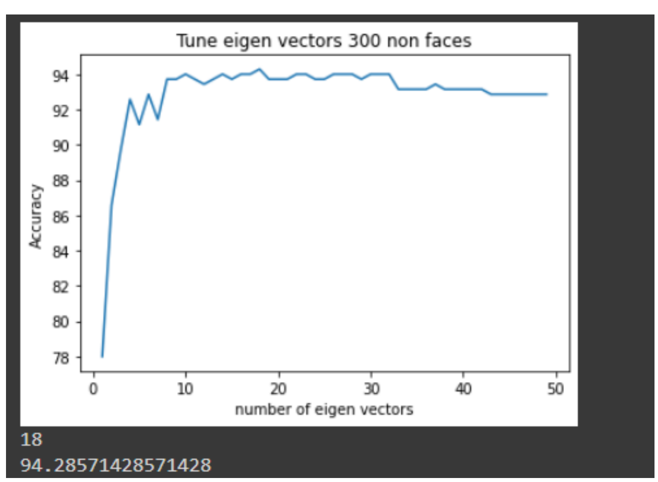
2. **How many dominant eigenvectors will you use for the LDA solution?**
    a. Non faces 100



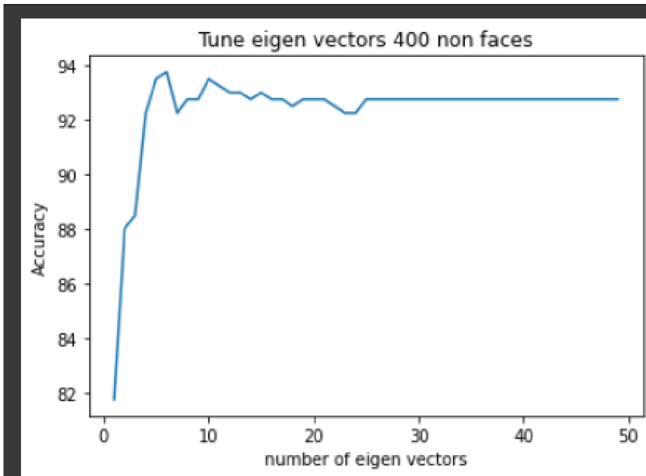
    b. Non faces 200



    c. Non faces 300

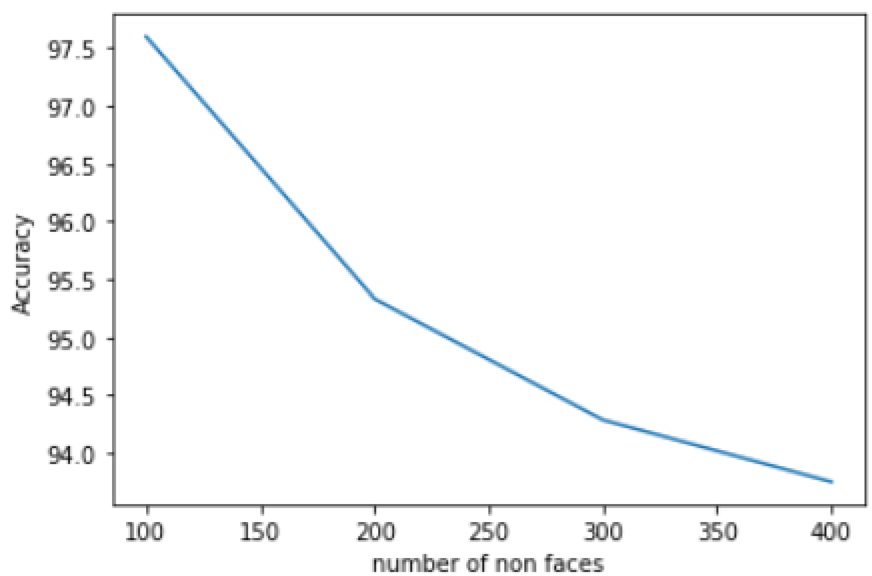


d. Non faces 400




```
6
93.75
```

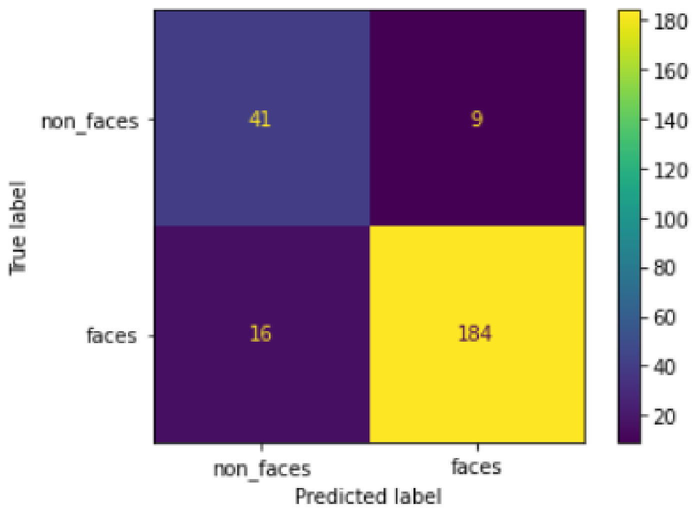
3. Accuracy while fixing faces at 400 image (Accuracy At Y -axis and number of images of non faces at X-axis)

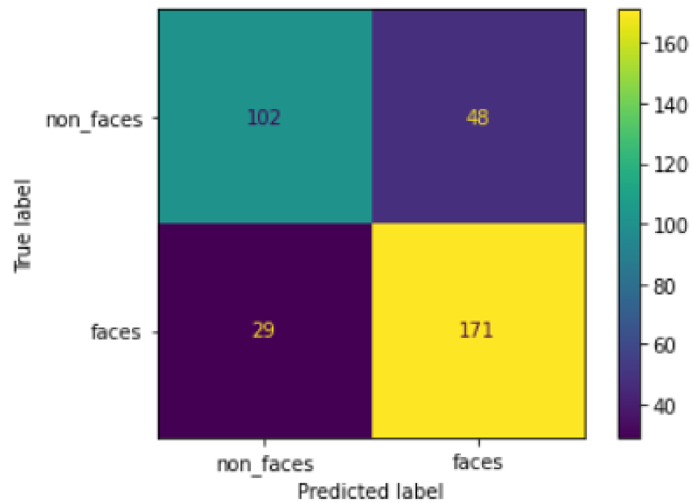


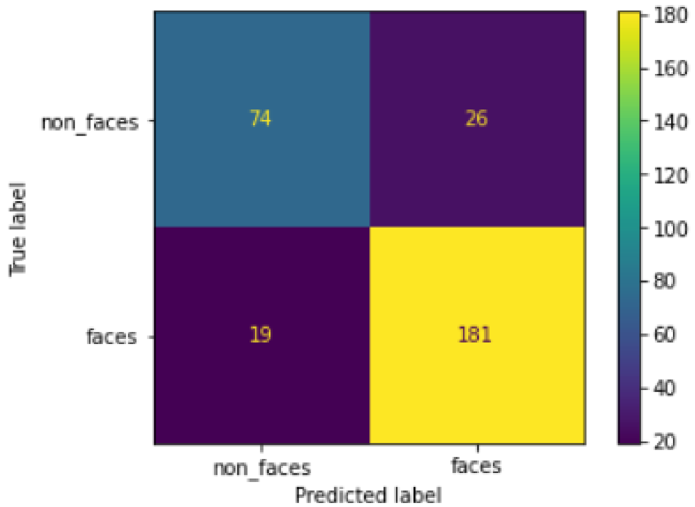
When taking eigen vectors (# classes -1 which equals 1)

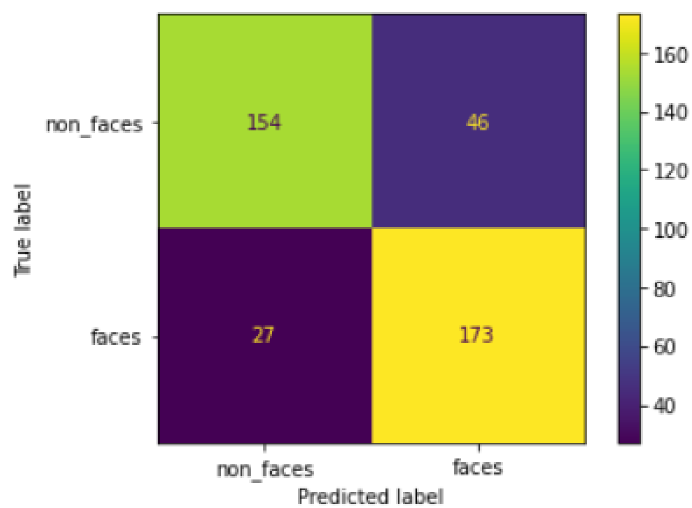
Success and failure cases while fixing faces at 400 images

Non facing equals 100, 200, 300, 400 in order

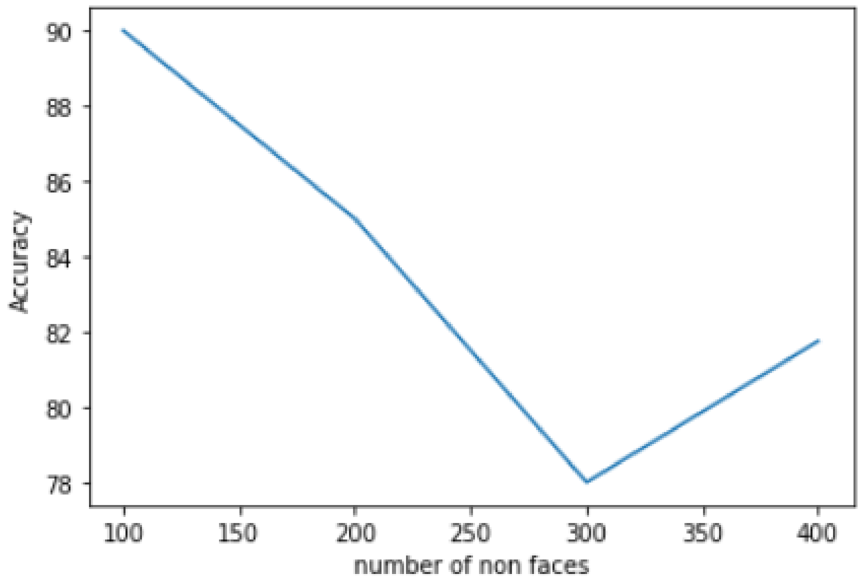








Accuracy while fixing faces at 400 image (Accuracy At Y -axis and number of images of non faces at X-axis)



NOTE

In normal accuracy should be high when data set of non faces is small as there is dominant class (faces) the it is decreasing until data is balanced then it start to increase again until some point that will lead to miss classify faces and over fitting.

But this approach like the above picture doesn't happen always as we see.

Bonus

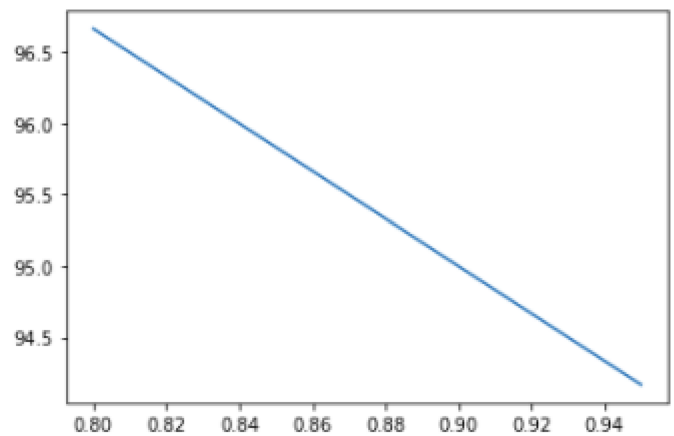
1. Data splitting (70 - 30)
    a. We loop over the images of every person(10 image per person) and take 7 images for training and 3 for testing and so on for every folder
    b. New training & testing list are passed to functions

2. Classification using PCA

   a. Accuracy summary for every alpha :

Alpha	0.8	0.85	0.9	0.95
Accuracy	96.7 %	95.8 %	95 %	94.167 %

   b. Relation between accuracy and alpha :

      i. 

      ii. There is a negative correlation between accuracy and alphas where if alpha increase the accuracy decrease

3. Classification using LDA

   a. Accuracy : 96.66%

   b. The accuracy increased as the training data increased to that level that at the same time doesn't lead to overfitting

   c. 70 30 is the normal split of data in machine learning