Assignment 1

Name:	**ID:**
Ahmed Osama Sakr	5463
Seif Eldin Amr	5481

Face Recognition

We intend to perform face recognition. Face recognition means that for a given image you can tell the subject id. We used the ORL dataset and downloaded it from Kaggle.

1. Discovering the Dataset and forming the Data matrix:

The dataset has images of 40 subfolders each folder is named ‘s1’ to ‘s40’ each subfolder contains 10 images of one person, to make the total number of images in the dataset 400 grayscale images of size 92x112.

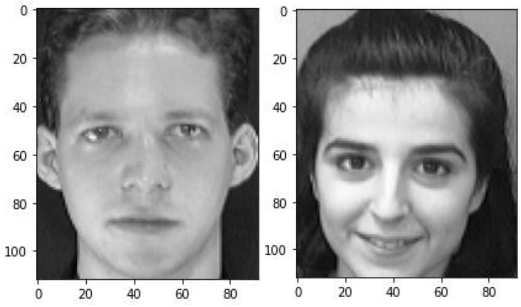


Image of label 1 Image of label 10

After reading the images, we flattened them to make the size of each image 1x10304, so at the end the data matrix that contained the dataset had a size of 400x10304.

To have labels for the images we took the substring of the subfolder name ‘s1’ to ‘s40’ that contained a number so, for ‘s1’ we got 1, for ‘s10’ we got 10, ect. The labels array was a 400x1 column vector with every label corresponding to the correct flattened image in the data matrix.

2. Splitting the Dataset into Training and Test sets:

As per the assignment handout’s instructions, we made a 50-50 split where the odd rows were kept for training and the even rows were kept for testing and we split the labels vector accordingly. This did give 5 instances per person in the training set and the test set. At the end we had a training data matrix of size 200x10304 and test data matrix of size 200x10304.

3. Classification using PCA:

Following the Pseudo Code of Principal Component Analysis, we computed the reduced dimensionality data matrix “A_train” and “A_test” for each alpha value that was required and then started classification using Sklearn’s K-nearest-neighbor classifier with uniform weights. Where we trained the classifier on “A_train” and then tested the classifier on “A_test” to get the following accuracies for each alpha. We tried used 50-50 split and using 70-30 split.

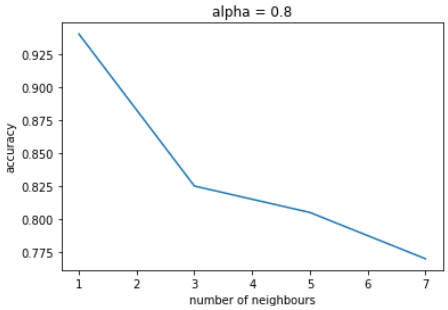
K = 1		
Alpha	Accuracy 50-50	Accuracy 70-30
0.8	0.94	0.9667
0.85	0.945	0.9583
0.9	0.94	0.95
0.95	0.935	0.934167

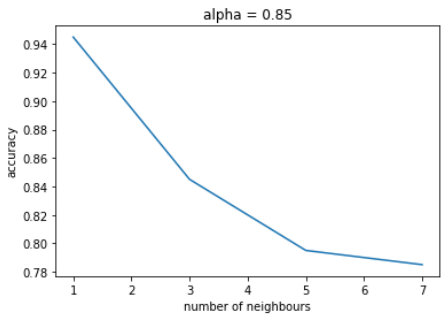
K = 3		
Alpha	Accuracy 50-50	Accuracy 70-30
0.8	0.825	0.9
0.85	0.845	0.9
0.9	0.85	0.9083
0.95	0.855	0.9

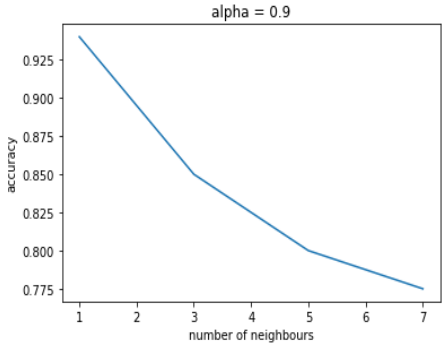
K = 5		
Alpha	Accuracy 50-50	Accuracy 70-30
0.8	0.805	0.875
0.85	0.795	0.9
0.9	0.8	0.875
0.95	0.79	0.8667

K = 7		
Alpha	Accuracy 50-50	Accuracy 70-30
0.8	0.77	0.81667
0.85	0.785	0.8083
0.9	0.775	0.8083
0.95	0.775	0.79166

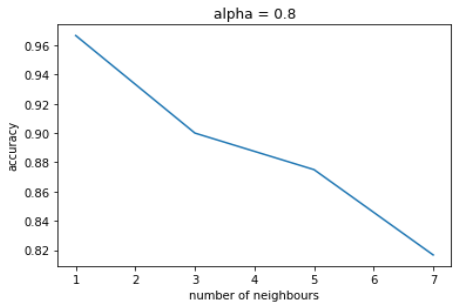
50-50 splits:

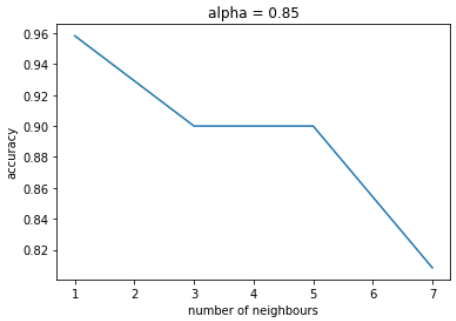


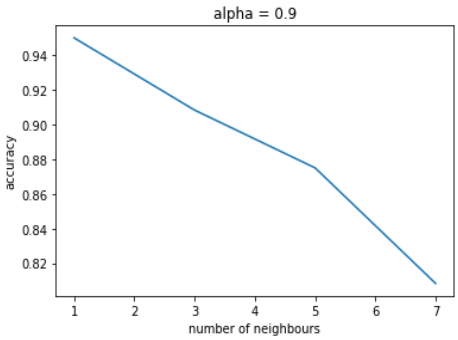


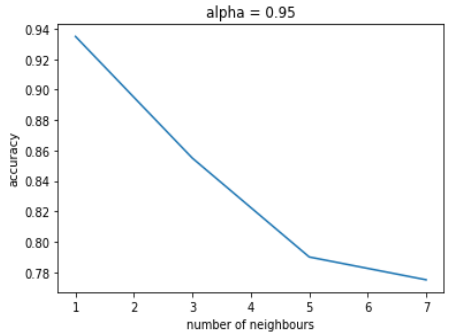


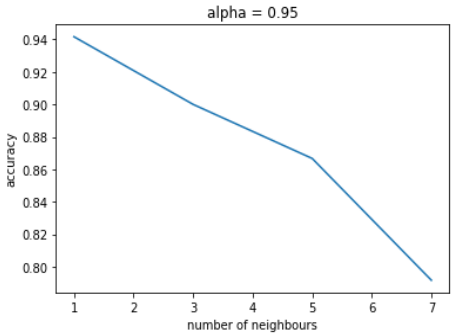
70-30 splits:







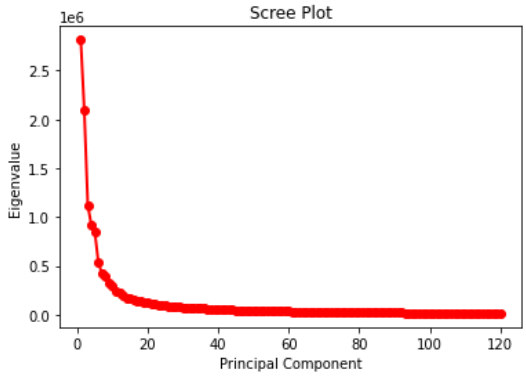




As we increase alpha, we increase the number of principal components that we use:

Alpha	PCs
0.8	36
0.85	52
0.9	76
0.95	117

And according to this scree plot, that shows only the first 120 PCs, the more principal components we use the smaller the eigen-values of this component get and in turn the smaller the variance contributed by this component.

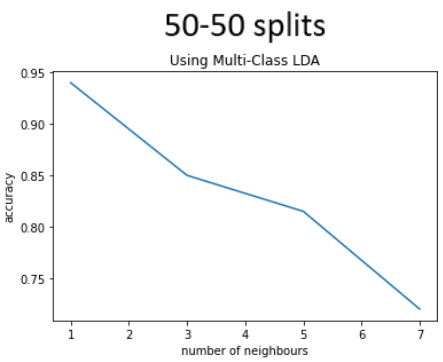


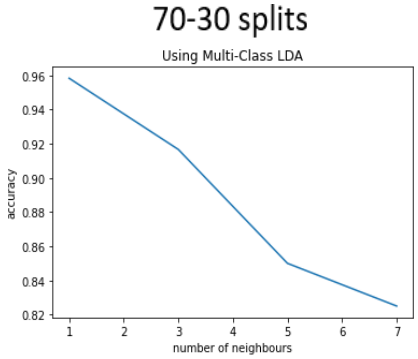
So, as we saw in the accuracies. The increase in alpha didn't do much to help improve the accuracies.

4. Classification using LDA:

Modified that given Pseudo Code of LDA the accommodate more than 2 classes. Firstly, making a function that returns the mean vector for every class and stack them into one matrix of size (n, m) where (n) is number of attributes of the data and (m) is the number of class labels. Then, computing between class scatter matrix (S_b), which is an (n, n) matrix where (n) is the number of attributes. Then, computing the within class scatter matrix (S). Finally, Running the normal LDA algorithm, using the first 39 principal components, and getting the projected instances of the training data "A_train" and the projected instances of the testing data "A_test" and putting it into the KNN classifier to get the classification accuracies. We tried used 50-50 split and using 70-30 split.

K	Accuracy 50-50	Accuracy 70-30
1	0.94	0.958334
3	0.85	0.91666
5	0.815	0.85
7	0.72	0.825





As the number of neighbors (k) increase the accuracy of LDA is better than that of PCA (except for k = 7), because the objective function of LDA is to maximize the distance between the projected means while decreasing the variance of each class.

Comparing between 50-50 and 70-30 splits:

The accuracies improved on all fronts with the PCA algorithm and with the LDA algorithm, and still LDA remains better as K increases.

5. faces vs non-faces images:

We used the natural images dataset from Kaggle[1], which is a compiled dataset of 6899 images from 8 distinct classes. But we only used from it, images of flowers, fruits, cars, motorcycles and airplanes, got a total of 600 images and put them on a bulk resizer[2] to resize them to 92x112.

Formed the data matrix that contains face and non-face images and the label vector where label 1 means face and label 2 means non-face.

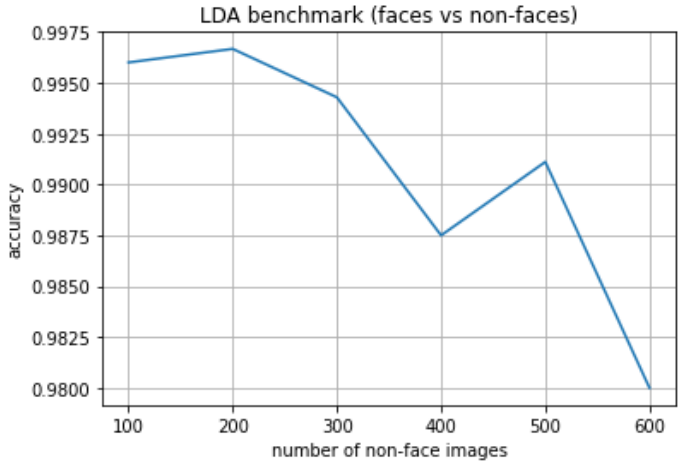
Ran the LDA algorithm on the new data matrix with changing the number of non-face images while keeping the number of face images constant.

We did LDA with (100, 200, 300, 400, 500, 600) non face images while keeping the number of face images 400. And then used the KNN classifier with k = 3.

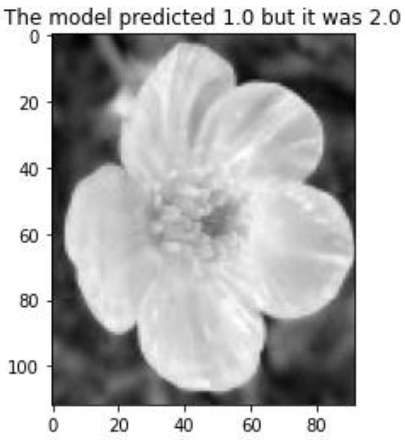
1 dominant eigen vectors was used for LDA. Because if we have (n) classes then in LDA the number of dominant eigen vectors that will be used is (n-1).

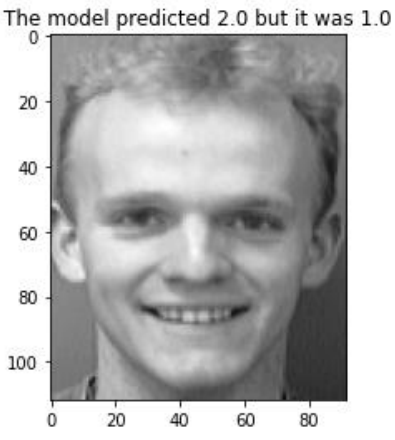
[1] Natural Images | Kaggle
[2] Bulk Resize Photos - Resize Images

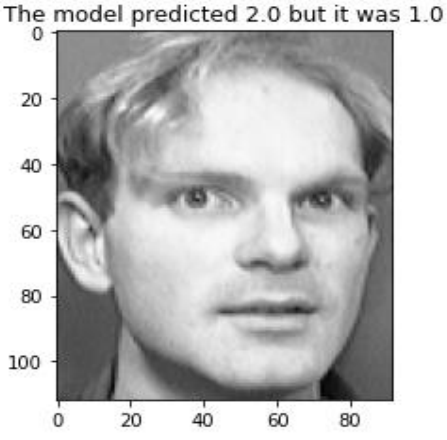
Number of non-faces images	Accuracy	Number of wrong predictions
100	0.996	1
200	0.99667	1
300	0.9942857	2
400	0.9875	5
500	0.99112	4
600	0.98	10



Some failure cases for LDA:

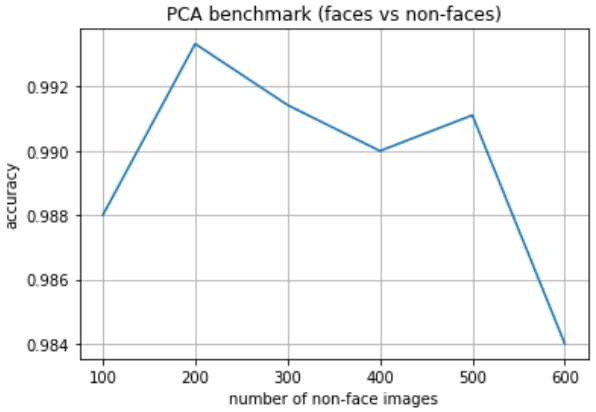




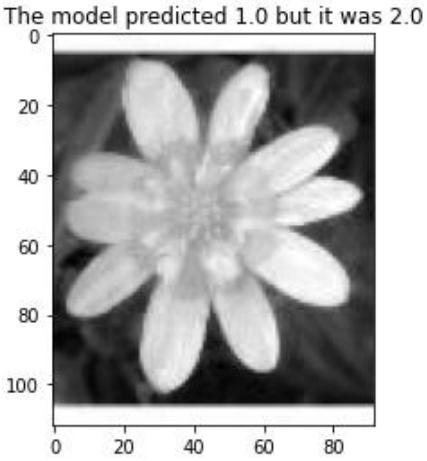


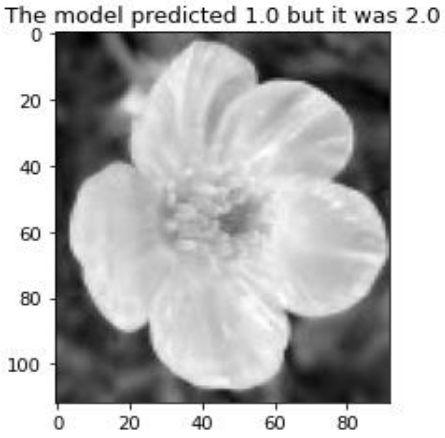
Then Ran PCA with the same format.

Number of non-faces images	Accuracy	Number of wrong predictions
100	0.988	3
200	0.9933	2
300	0.99143	3
400	0.99	4
500	0.9911	4
600	0.984	8



Some failure cases for PCA:





For LDA, as we increased the number of non-face images in the data, the accuracy kept decreasing. And this is probably due to the increase in the variance of the non-face images as we kept adding them.

For PCA increasing the number of non-face images increases the accuracy because as we did increase the number of non-face images the number of wrong predictions kept being almost constant until 500 non-face images and then began to decrease as the number of non-face images surpassed the number of face images.