Fundamentals of Statistical Learning
CSE569

Project Report - Discriminant analysis for recognition of human face images

Challenge: Creating a classifier that can classify a test image into one of the classes that belong to the training set.

Data Set: Consists of 300 '.png' images in total. There are 10 images for every person and there are 30 such people. Each image size is 128x128 pixels.

Concept: Linear Discriminant Analysis is used to solve the above challenge. LDA provides a methodology to reduce the total scatter within a class and increase the scatter between classes, which in turn can be used to build an effective classifier.
Assuming, initially the classes in training data set are not distinguishable, once LDA is performed on this training data set, it produces certain Eigenvectors onto which if the training set is projected, the classes become distinguishable and a new test dataset can then be easily classified by projecting it onto the direction of the Eigenvectors (same as that of the training set) and using any of the distance formulae (Euclidean/KNN), we can then find out which is the closest class that the test set belongs to amongst the training set on the projected plane.

Method/Procedure:
Programming tool used Matlab.

Programming Steps:

- Initial size of image is a matrix of size 128x128x3, which corresponds to the rgb intensity values of the image.
- For all calculations the gray scale value of the image is used, it reduces the dimension of each of the images to a 128x128x1 matrix.
- Since carrying out matrix computations on a 128x128 matrix is time consuming on an averagely configured pc, I have resized the images and performed computation for a 40x40 matrix, 64x64 matrix and 96x96 matrix.
- Each of the images is then represented as vector forms, reshaping the original image matrix performs this task. Now, each image is represented by a 1600x1 vector(for a 40x40 size image).
- We need the means of each of the classes to compute the within class scatter matrix. Once the mean is computed for that particular class we compute the within class scatter by summation Pr(C)*(x-m)*(x-m)' for every x in the set. (x = each of the image in the training set, m = mean of the images)
- To calculate the total within class scatter we add all the within class scatter matrices from all the classes.
- Now to calculate the between class scatter we need the mean of all the classes and the total mean of the 300 classes. The between scatter matrix is formed by summation Pr(C)*(m-m`)*(m-m`)' for each class(m = mean of that class, m` = mean of all the classes).
- Eigenvalues and corresponding Eigenvectors are now computed from the equation w=inverse (Sw) * Sb (Sw = Scatter within class, Sb = Scatter between class).

- Project the training images onto the set of Eigenvectors (different number of Eigenvectors are taken to check for varying accuracy).
- Project the testing images onto the same set of Eigenvectors.
- Now we have our projected training set and our projected test set.
- We pass these two sets to a Euclidean/KNN classifier to get the accuracy of our model.

Results and Conclusions:

a) For image resized to 40x40

Accuracy Table

No. of Eigenvectors	Percentage of correctly classified test sets
1	10
2	46.67
3	76.67
4	80
5	90
6	83.33
7	90
8	86.67
9	96.67
10	96.67
19	96.67
20	100
29	100

b) For image resized to 64x64

Accuracy table

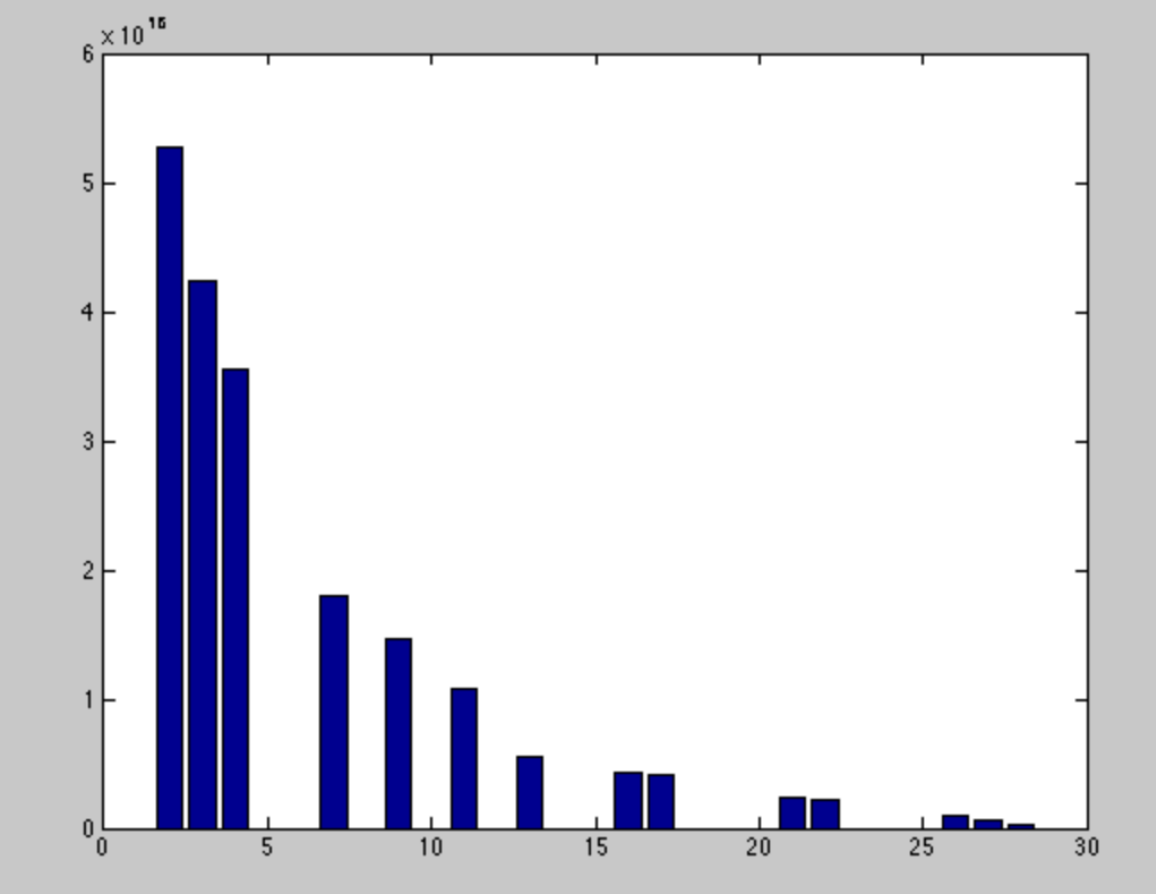
No. of Eigenvectors	Percentage of correctly classified test sets
1	13.33
5	63.33
10	93.33
15	100
29	100

c) For image resized to 96x96

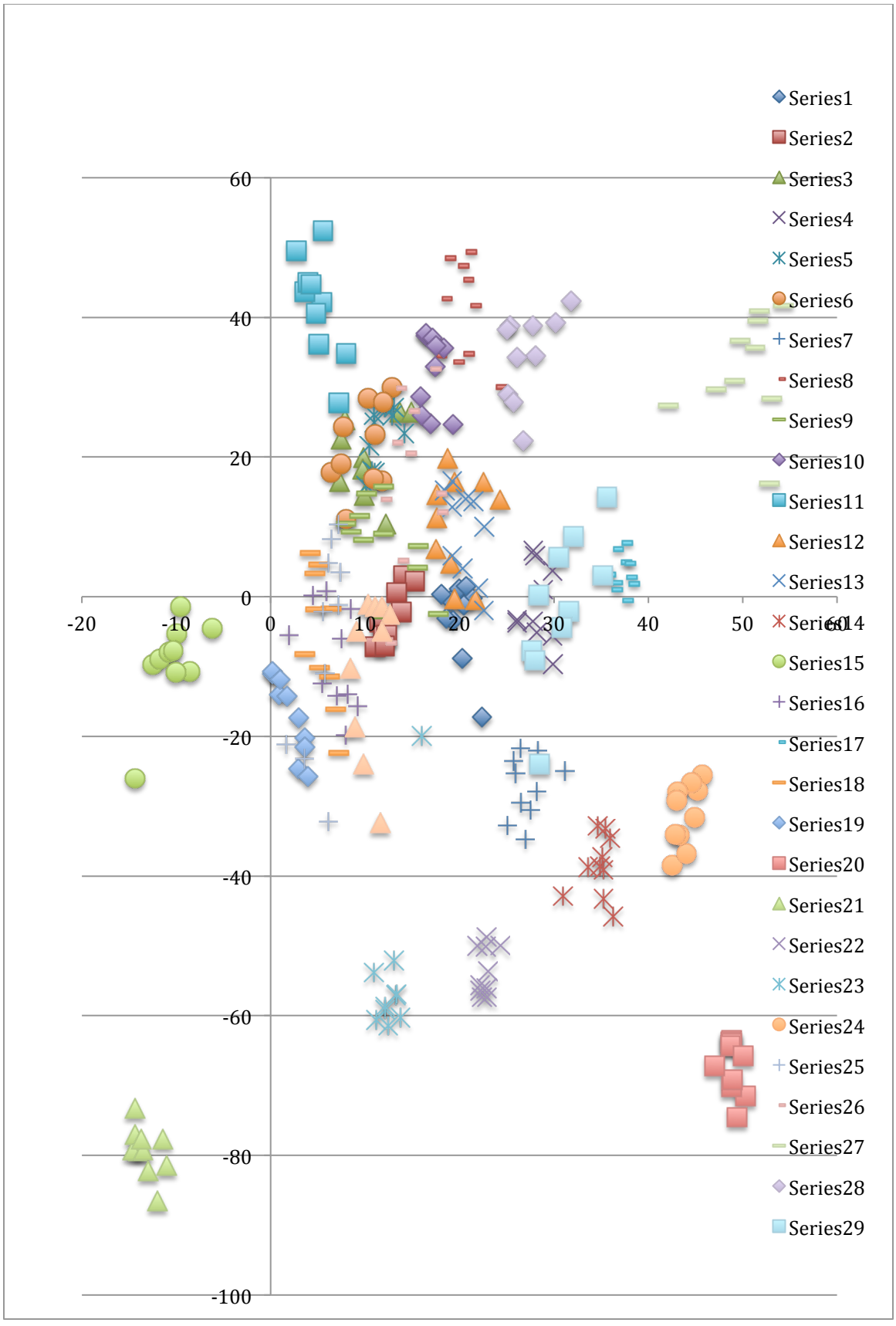
Accuracy table

No. of Eigenvectors	Percentage of correctly classified test sets
1	13.33
5	86.67
10	93.33
15	100
29	100

Below is a graph representing the discriminating power of top 29 selected Eigenvectors for a resized image 40x40.



Below is a graph representing a scatter plot for each of the classes in our training set after being projected onto two-dimensional feature space. The result obtained is for resized images (40x40). As we can see visually that there are nearly 30 clusters formed, though there is a fair overlap between clusters. From the above accuracy table for a resized 40x40 image the accuracy of classification is 46.67 when the number of Eigenvectors taken is 2.


60
40
20
0
-20
-40
-60
-80
-100
-20
-10
0
10
20
30
40
50
60
Series1
Series2
Series3
Series4
Series5
Series6
Series7
Series8
Series9
Series10
Series11
Series12
Series13
Series14
Series15
Series16
Series17
Series18
Series19
Series20
Series21
Series22
Series23
Series24
Series25
Series26
Series27
Series28
Series29

Conclusion:

- LDA classifier successfully classifies the test case into one of the training sets. From the accuracy table it is evident that after taking certain number of eigenvectors the accuracy reaches a certain threshold and thereafter does not change even if the number of eigenvectors taken for projection is increased.

- Only the first C-1 eigenvectors affect the accuracy of the classifier (where C is the number of classes), any number of eigenvectors taken after this point will not affect the accuracy.

- We need a high configuration system to compute LDA for the entire image (128x128).

- The eigenvectors found from the Rayleigh coefficient are the directions in which the projections of the data set should take place in order to attain maximum distinguishability between classes.

How to run the Matlab Code

Code consists of 6 functions(6 files) –

1) main
2) read_data
3) projected
4) scatter_within
5) scatter_between_total
6) get_test_data

main – computes the within scatter for all classes, between scatter, finding out eigenvectors and eigenvalues, projection of images(test and training) and classification using KNN and finally computes the accuracy of the classification.

read_data – returns a matrix to the main function that consists of all the images for 1 person in a vector form. Therefore it consists of 10 columns and rows depend on the size of the matrix after resizing (incase of 40x40, rows are 1600).

Projected – computes the projection using eigenvectors and the training/test data sets, returns the projected set to the main function. Parameters it takes is the set of all training images and the set of Eigenvectors computed.

Scatter_within – computes the scatter within every class. Parameters it takes is the set of training images for a particular person and the mean for that particular class.

Scatter_between_total – returns the value of the total between scatter to the main function, parameters it takes is the mean of all the training sets.

Get_test_data - returns a matrix to the main function that consists of all the test images for all person in a vector form. Therefore it consists of 30 columns and rows depend on the size of the matrix after resizing (incase of 40x40, rows are 1600).

Output – 1) dimension of projected test data set
2) dimension of projected training data set
3) accuracy of classification of test data set
4) classified test cases