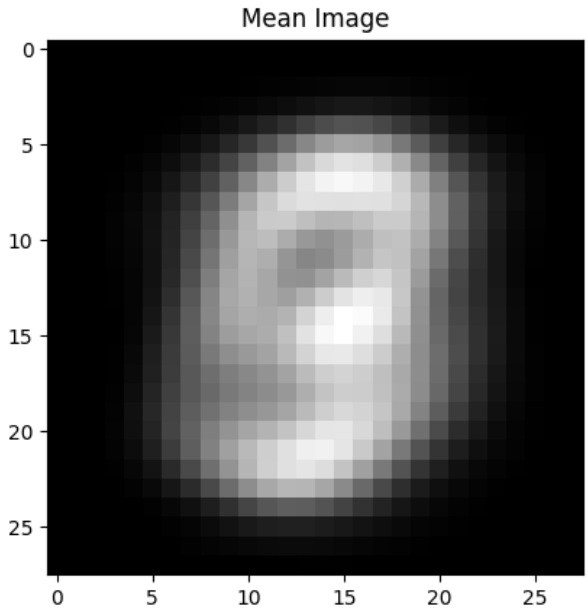
SML Assignment 2

-Harkishan Singh(2017233)

Question 1 :

a) We are required to compute mean and covariance of the data. Data contains 60k training images and 10k testing images. I have taken training images and at first I have vectorised the images by flattening them (images are of 28*28 pixels). By vectoring them, I have a 2-d array of dimension (60000,784). Now computing mean of these 60k vectors is a very easy task. Just add up all the vectors and dived them by 60000. Mean image will somewhat look like :

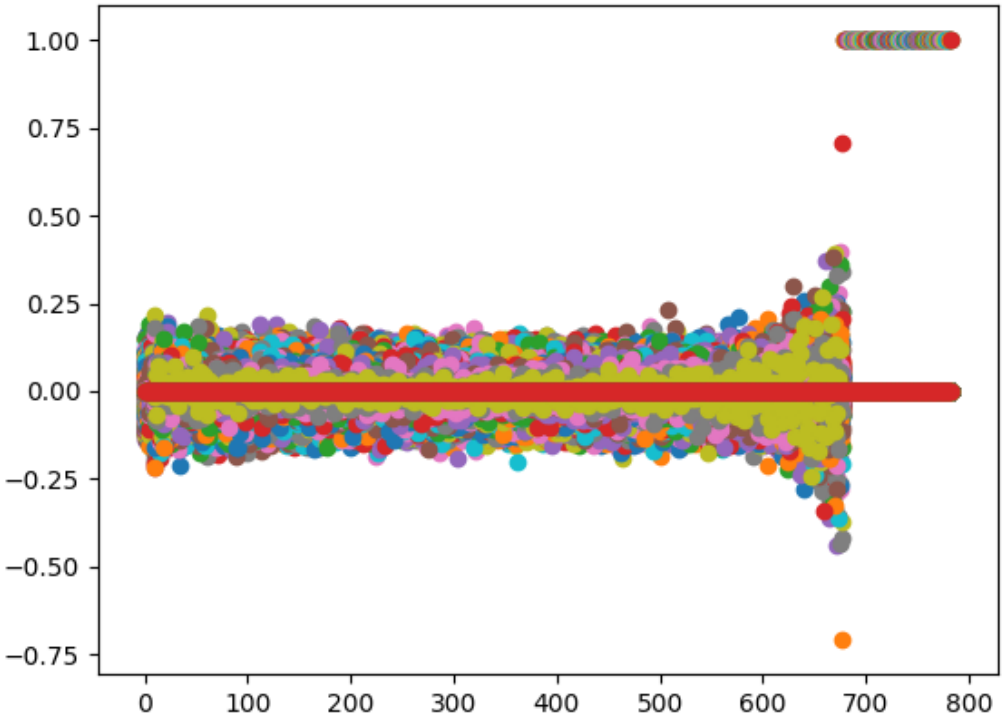


For computing covariance matrix, I have looped over 60000 vectors and adding a term of : (x-m)(x-m)transpose, and then when loop is executed, I divided the covariance matrix by 59999. Dimension of covariance matrix will be : (724,724).

b) For PCA : I have computed eigenvalue and eigenvectors of the covariance matrices using Numpy.linalg.eig() function and that is we are required to do in this part for FDA, first I have computed in-class-covariance matrix and the find out the eigenvalues and eigenvector of the respective matrix which we take in account for FDA.

c) For this part I have simply plotted the result of above part using marplot library, a view will look like :

Next Page =>



d) Using the FDA function from part b), its discriminant function can be easily computed as we have both in-class-covariance matrix and the overall covariance matrix.

e) For doing this part, I have first sorted the eigenvalue list, such that, we have the highest value eigenvalue and eigenvector at first place and then, we can just iterate over the list to get the required eigenenergy (=95%)

f). PCA is already done in part b, we just have to print out the eigenvectors are images.

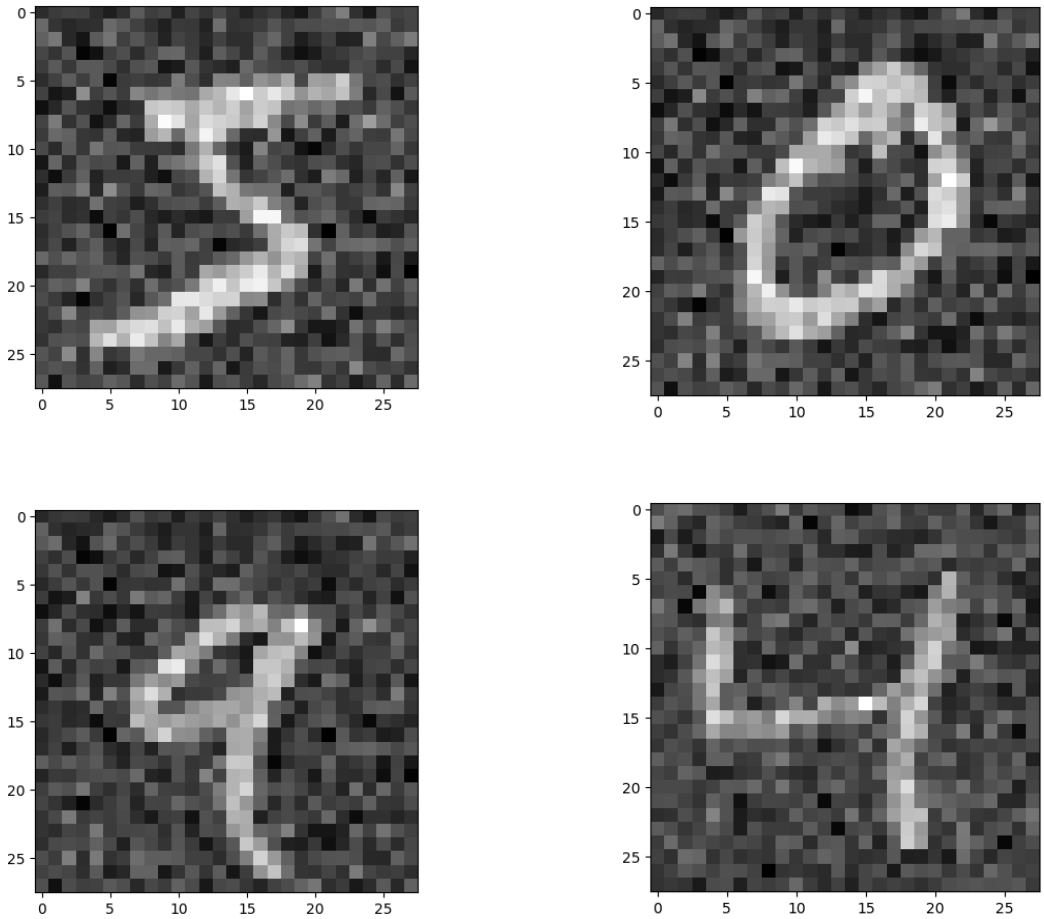
g) Similar to the part e), I have iterated over the eigenvalue list and wherever I have got the required eigenenergy, I break the loop : for 70%, we have 7 eigenvectors, for 90% we have 8 eigenvectors and for 99%, we have 10 eigenvectors.

h) For doing this part, we'll just iterate over the testing images and just compare the trained images label and just compute the accuracy by looking at what is the result and what is predicted by the program.

Next Page =>

Question 2 :

For this question, we are required to add gaussian noise to the images. For doing that, I have constructed a function which returns me a matrix of dimension (28*28) in which the elements contains random variables drawn from gaussian distribution and I have simply added this to the images. A sample will look like :



For removing the noise from the data, I have implemented PCA, by computing eigenvectors and eigenvalues, I have taken top 7 eigenvector and then I have reversed those eigenvector to form the original image (X).

Original image looks like :