Question 1

To load the data, use the “pickle” library. This decodes the data into a large dictionary with labels and data.

To plot the data, use the plt.imshow() function. This needs data in the form of an np.array with dimensions (3,32,32) for an image with RGB values and a 32x32 image.

To calculate the LDA using sklearn dataset, simply use the lda.fit() and lda.predict() functions. lda.fit() takes in the labels and the data as input and creates a linear division rule to map the test data to classes.

lda.predit() is used to map the test data into classes.

Towards the end, simply calculate the correct number of mappings to deduce the accuracy of the analysis function.

Question 2

To decode the data, I have used https://github.com/Ghosh4AI/Data-Processors/blob/master/MNIST/MNIST_Loader.ipynb as a reference. Essentially, if the magic number is 2051, it is data to be loaded. If it is 2049, then it is the label number.

To calculate the PCA, use sklearn PCA(n_components) pca.fit() and pca.transform() functions. The fit() function learns the data. The transform() function reduces the dimensions of the data, and is used to fit transform an n x m matrix to an n x n_components matrix.

Now, calculate the LDA as explained in the previous question. Use n_components = 15, 8, 3. The accuracy reduces as the number of components reduces. As the LDA function has lesser data to analyze due to PCA reducing data, the accuracy becomes lesser and lesser.

Question 3

Read the data using the read_csv command from the pandas library. To calculate the FDA, proceed sequentially.

1. Divide the data classwise
2. Calculate the classwise means
3. Compute the Si matrices. Use the formula $(x_i\text{-}Mu_i)(x_i\text{-}Mu_i)^T$. Do this classwise.
4. Compute the Sw matrix by summing up all the S_i matrices.
5. Compute the overall Mean, Mu.
6. Compute the eigenvectors of $Sw^{-1}Sb$
7. Let W be the first c-1 components of $Sw^{-1}Sb$
8. Compute LDA on $Y = W^TX^T$ as dataset and use predict on $W^T(\text{testing data})^T$
9. Simply compute classwise accuracy as done previously.

Question 4

Use common logic and perform the same steps from questions 1, 2, and 3. A key point to remember is that the MNIST dataset does not have equal data points per class. So, we will need to make minor changes in code to ensure correctness.

Question 5

a. In case we have the sign function as our activation function, we need to check if

$sgn(\beta_0 + \beta x_i) = y_i$. Here $y_i \in \{-1, 1\}$

If the condition is true, we say that the point is correctly classified. Else, we say that it is not correctly classified.

Now, if the point is not correctly classified, we say want to minimize the distance function, d_i. This function is given by

$d_i = -y_i(\beta_0 + \beta x_i)$

We perform this by the gradient descent method taught in class. But essentially, the function we need to minimize is d_i.

If we have a sigmoid function, we know that the activation function needs to output binary values for a Rosenblatt's perceptron. Therefore, we need to modify the usage of the sigmoid function for a Rosenblatt's perceptron. Let us use the following rule to classify.

If $z_i \geq 0.5 \; and \; sigmoid(\beta_0 + \beta x_i) \geq 0.5 \Rightarrow correctly\ classified$

If $z_i < 0.5 \; and \; sigmoid(\beta_0 + \beta x_i) < 0.5 \Rightarrow correctly\ classified$

Else, incorrectly classified.

Now, if the point is not correctly classified, we say want to minimize the distance function

$d_i = -y_i(\beta_0 + \beta x_i)$

We know that $y_i \in \{-1, 1\}$. Therefore, we need to map $z_i \; to \; y_i$ such that their ranges are same.

We can map this using, $y_i = sgn(2z_i - 1)$. Also assume $sgn(0) = 1$.

In essence, the function d_i will remin the same. Hence, the update rule parameters do not change even if the activation function is changed.

Date
Page

SML Assignment 3.

Qs. b $\phi(\beta, \beta_0) = -\sum_{i=1}^{N} y_i [\beta^T x_i + \beta_0)$.

We know $\sqrt{(\beta^T \beta)} = 1$

$\Rightarrow \beta^T \beta = 1$

As we have a constraint and an expression to minimize, let us use the lagrangian to minimize.

$\Delta \phi(\beta, \beta_0) - \lambda(\Delta \beta^T \beta - 1) = 0$

$\Rightarrow -\sum_{i=1}^{N} y_i (x_i) - \lambda 2\beta = 0$

$\Rightarrow -\sum_{i=1}^{N} y_i - \lambda(0) = 0.$

$\therefore -\sum_{i=1}^{N} y_i x_i = 2\lambda\beta$

and $\boxed{\sum_{i=1}^{N} y_i = 0.}$

Now, let us calculate β.

We know, $\beta^T\beta = 1$

$$\therefore \left(\frac{-\sum_{i=1}^{N} y_i x_i}{2\lambda}\right)^T \left(\frac{-\sum_{i=1}^{N} y_i x_i}{2\lambda}\right) = 1$$

$$\Rightarrow \left(\sum_{i=1}^{N} y_i x_i\right)^T \left(\sum_{i=1}^{N} y_i x_i\right) = 4\lambda^2$$

$$\text{or} \quad \left(\left\|\left(\sum_{i=1}^{N} y_i x_i\right)\right\|_2\right)^2 = 4\lambda^2$$

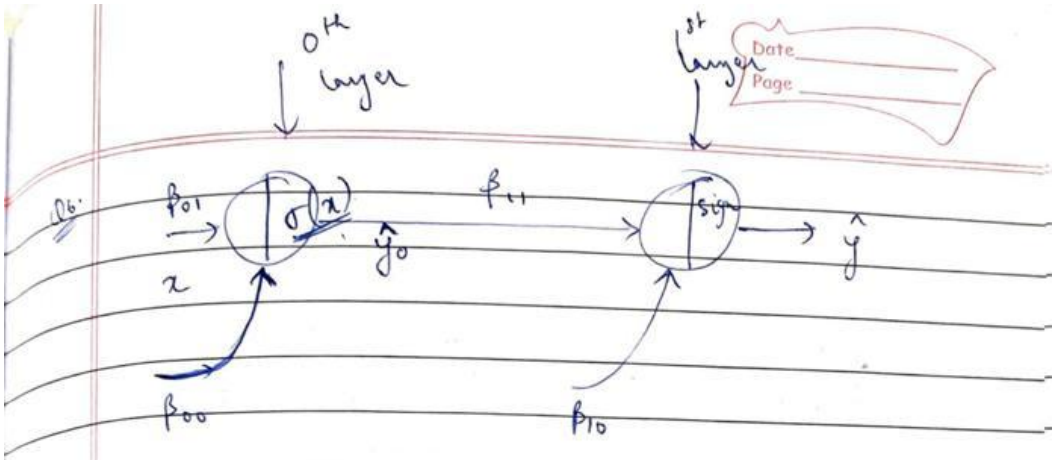
$$\text{or} \quad \boxed{\left\|\sum_{i=1}^{N} y_i x_i\right\|_2 = 2\lambda}$$

$$\therefore \boxed{\beta = \frac{-\sum_{i=1}^{N} y_i x_i}{2\left\|\sum_{i=1}^{N} y_i x_i\right\|_2}}$$

Therefore, the update rule can be represented as:-

$$\begin{pmatrix} \beta_{new} \\ \beta^0_{new} \end{pmatrix} = \begin{pmatrix} \beta_{old} \\ \beta^0_{old} \end{pmatrix} - \eta \nabla \left(\frac{\partial \phi(\beta, \beta_0)}{\partial} \right)$$

$$\Rightarrow \boxed{\begin{pmatrix} \beta_{new} \\ \beta^0_{new} \end{pmatrix} = \begin{pmatrix} -\sum_{i=1}^{N} y_i x_i \Big/ 2\left\|\sum_{i=1}^{N} y_i x_i\right\|_2 \\ \beta^0_{old} \end{pmatrix} + \eta \begin{pmatrix} \sum_{i=1}^{N} y_i x_i \\ 0 \end{pmatrix}}$$



Where σ is the sigmoid f^n and sign is the signum function

$$\hat{y}_0 = \sigma(\beta_{01} x + \beta_{00})$$

$$\hat{y} = \text{sign}\left(\beta_{11}(\sigma(\beta_{01} x + \beta_{00})) + \beta_{10}\right)$$

$$\beta_{new} = \beta_{old} + \eta \frac{\partial E}{\partial \beta}$$

Here, $E = d = -y(\beta_{11}\hat{y}_0 + \beta_{10})$

$$= -y(\beta_{11}(\sigma(\beta_{01} x + \beta_{00})) + \beta_{10})$$

$$\frac{\partial d}{\partial \beta_{11}} = -y \cdot \sigma(\beta_{01} x + \beta_{00})$$

P.T.O.

$$\frac{\partial d}{\partial \beta_{10}} = -y$$

$$\frac{\partial d}{\partial \beta_{01}} = -y\ \beta_{11}\ (\sigma(\beta_{01}x + \beta_{00}))\ (1 - \sigma(\beta_{01}x + \beta_{00}))\ x$$

$$\frac{\partial d}{\partial \beta_{00}} = -y\ \beta_{11}\ (\sigma(\beta_{01}x + \beta_{00}))\ (1 - \sigma(\beta_{01}x + \beta_{00}))\ .1.$$

Update as the following:-

$$\beta_{00}^{new} = \beta_{00} + \left[y\ \beta_{11}\ (\sigma(\beta_{01}x + \beta_{00}))\ (1 - \sigma(\beta_{01}x + \beta_{00}))\right]\eta$$

$$\beta_{01}^{new} = \beta_{01} + \left[y\ \beta_{11}\ x\ (\sigma(\beta_{01}x + \beta_{00}))\ (1 - \sigma(\beta_{01}x + \beta_{00}))\right]\eta$$

$$\beta_{10}^{new} = \beta_{10} + [y].\ \eta$$

$$\beta_{11}^{new} = \beta_{11} + \left[y\ \sigma(\beta_{01}x + \beta_{00})\right]\eta$$

where η is the learning parameter.