Report
Assignment 4

Assignment 4

(1) $y_k(x,w) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$

$$\sigma(a) = \{1 + \exp(-a)\}^{-1}$$

$$\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$$

$$\tanh(a) + 1 = \frac{e^a - e^{-a}}{e^a + e^{-a}} + \frac{e^a + e^{-a}}{e^a + e^{-a}}$$

$$\tanh(a) + 1 = \frac{2e^a}{e^a + e^{-a}} = \frac{2}{1 + e^{-2a}} \quad \text{— (1)}$$

we also know that, $\sigma(2a) = \dfrac{1}{1 + e^{-2a}}$ — (2)

From (1) & (2)

$$\tanh(a) + 1 = 2\sigma(2a)$$

$$\frac{1}{2}\tanh\left(\frac{a}{2}\right) + \frac{1}{2} = \sigma(a)$$

If $\sigma(a)$ is our activation function, then we can multiply all the input-hidden weights by $\left(\frac{1}{2}\right)$, making the input to the hidden layer as $\frac{a}{2}$.

→ If we scale each of hidden outputs by $\frac{1}{2}$, and add $\frac{1}{2}$ to the bias hidden output, we will get the small result.

→ Hence parameters of 2 networks differ by linear transformations.

(2) $\sigma(\theta_0 + x_1\theta_1 + x_2\theta_2)$, $w_i \in$ integer, $x \in \{0,1\}$

$$\sigma(k) = \begin{cases} 1 & \text{if } k \geq 0 \\ 0 & k < 0 \end{cases}$$

a) $\theta_0 = -30$

consider the following values of $\theta_1 = 20$ & $\theta_2 = 20$

$\sigma(-30 + 20x_1 + 20x_2)$ as the function.

The 4 possible values of x_1 and. x_2

x_1	x_2	$\sigma(-30+20x_1+20x_2)$
0	0	$\sigma(-30)\approx 0$
0	1	$\sigma(-10)\approx 0$
1	0	$\sigma(-10)\approx 0$
1	1	$\sigma(10)\approx 1$

This is exactly AND function

b) If we want NOR, we have to put large weights infront of $x1$, x_2. Consider the following values, $\theta_0 = 30$ $\theta_1 = -40$ and $\theta_2 = -40$

$\sigma(30-40x_1-40x_2)$

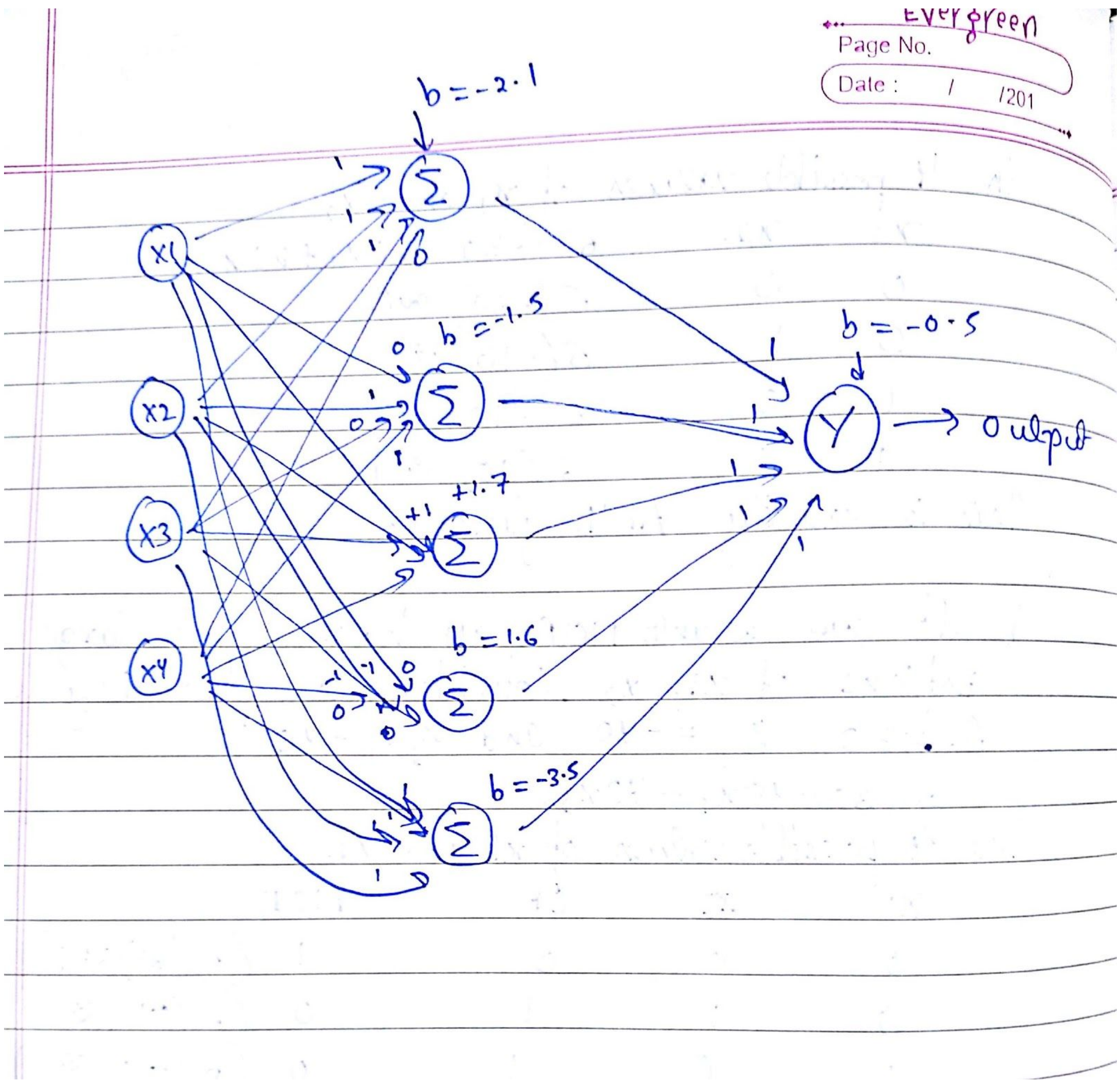
The 4 possible values of $x1$ and $x2$

X1	X2	OR	NOR
0	0	0	1 $(\sigma(30)\approx 1)$
0	1	1	0 $(\sigma(-10)\approx 0)$
1	0	1	0 $(\sigma(-10)\approx 0)$
1	1	1	0 $(\sigma(-50)\approx 0)$

This is exactly NOR function.

c) $(x_1 \wedge x2 \wedge x_3) \vee (x2 \wedge x4) \vee (x1 \wedge x4)' \vee (x2 \wedge x3)' \vee (x1 \wedge x2 \wedge x3 \wedge x4)$

- for sigmoid function, $y \geq 0.5$ if $(x \geq 0)$ and $y \leq 0.5$ if $x<0$
- For $(x_1 \wedge x_2 \wedge x_3)$ given the above statement, we need to choose w_i and ~~b~~ b s.t. $b + w_1x_1 + w_2x_2 + w_3x_3$ will be greater than 0 when $(x_1 \wedge x_2 \wedge x_3)$ is equal to 1.
- So one candidate solution is $1x_1 + 1x_2 + 1x_3 + 0x_4 + (-2.1)$
- Sly, we ~~can~~ can do for all the other parts. The final network is shown below:-



3)

Data set used - MNIST dataset

Activation function to be used - ‘tanh’

Neurons in 1st hidden layer - 500

Neurons in 2nd hidden layer - 250

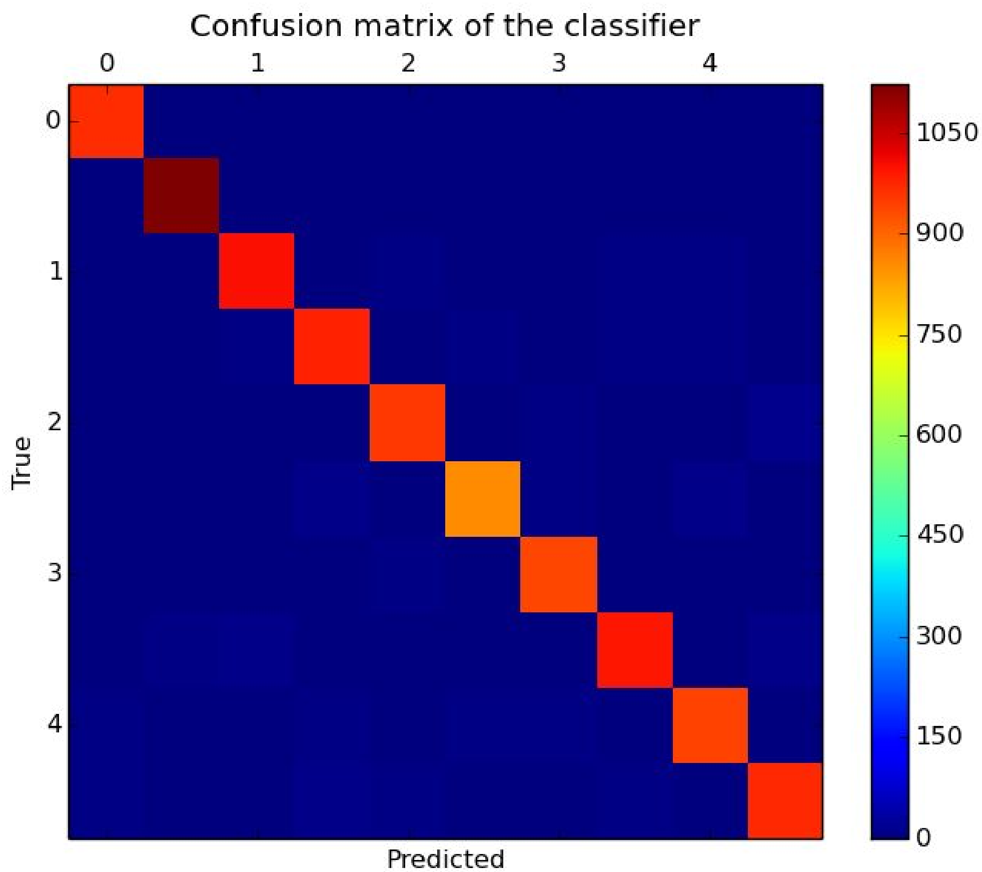
Steps followed:

- Converted ubyte file of MNIST dataset to .csv file.
- Installed nolearn for neural network.
- Used the MLPClassifier in sklearn to implement a multilayer neural network.
- Trained the model using the above mentioned parameters in MLPClassifier and predicted the outputs for the test data.

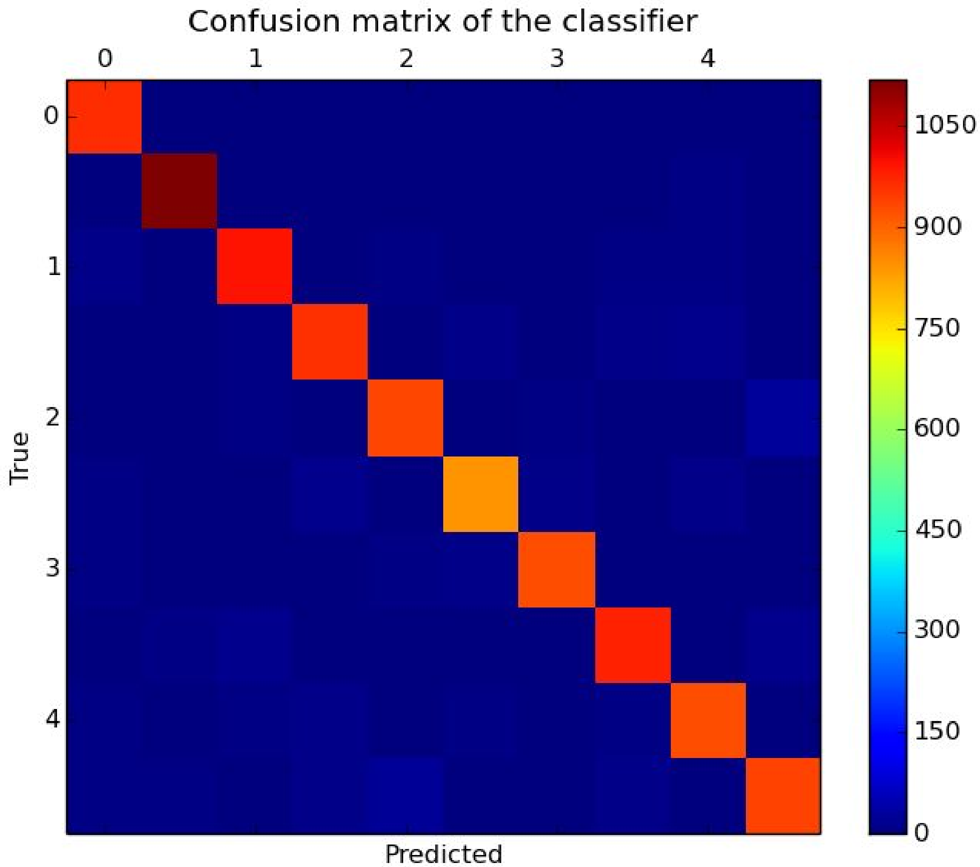
- Then computed the accuracy and made the confusion matrices for different learning rates.

Results for different Learning Rate values with their corresponding plot of confusion matrices are as follows:

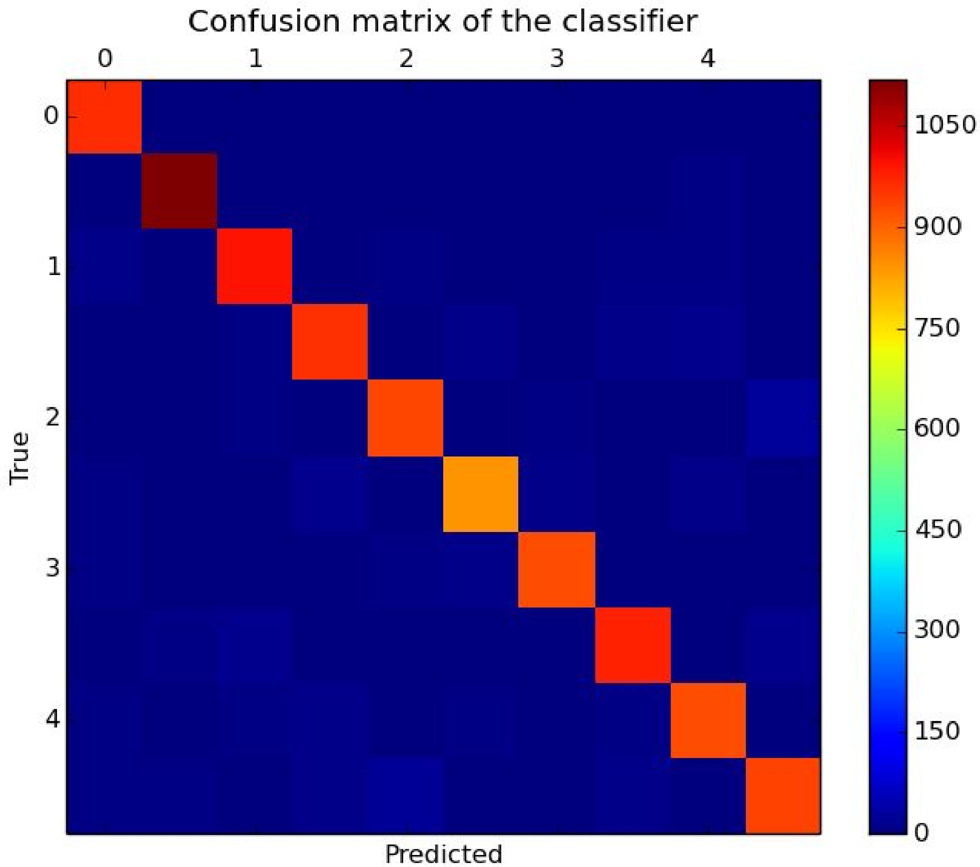
i)Learning Rate = **0.001**
Accuracy Score = **97.29%**



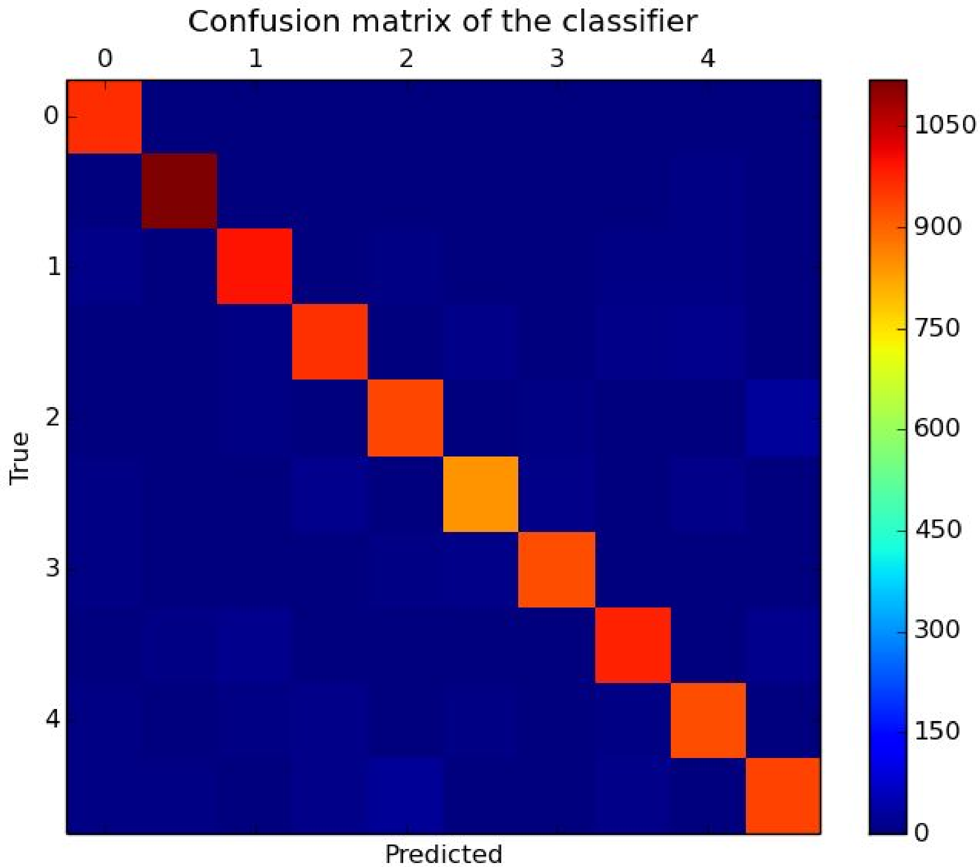
ii)Learning Rate = **0.01**
Accuracy Score = **96.89%**



iii)Learning Rate = **0.1**
Accuracy score: **95.80%**



iv)Learning Rate = **0.0001**
Accuracy score = **96.89%**



Learning Rate	Accuracy
0.1	**95.80%**
0.01	**96.89%**
0.001	**97.29%**
0.0001	**96.89%**

So, we can conclude that when learning rate increases from 0.1 to 0.001, the accuracy increases.

Question 4.

Auto encoder values :
No. of neurons in input layer - 784
No. of neurons in hidden layer - 100

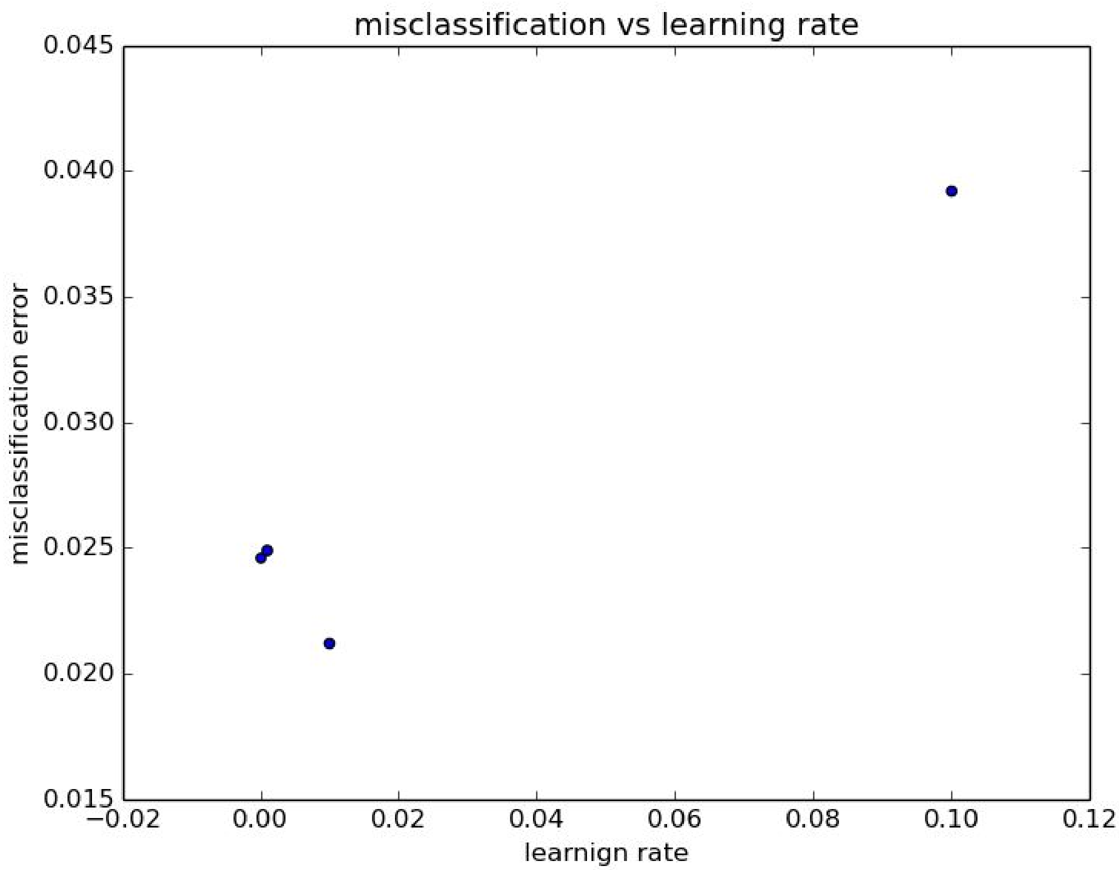
Steps followed:

- Divided the dataset into training and testing.
- Installed theano and keras
- Made an Autoencoder with input vector of size 784 neurons and hidden layer of size 100.
- Passed the output to the feed forward neural network with 100 neurons on input layer, 50 neurons on hidden layer and 10 neurons on output layer.

Visualization by the autoencoder

Feed Forward Neural Network values :
No. of neurons in input layer - 100
No. of neurons in hidden layer - 50
No. of neurons in output layer - 10

Learning Rate	0.1	0.01	0.001	0.0001
Accuracy	97.88 %	96.08 %	97.51 %	97.54 %



Conclusion

- The accuracy given by autoencoder are much higher than MLPClassifier for the same learning rates, hence we can say that autoencoder works better for MNIST dataset.