# ASSIGNMENT-3

# Q2- Artificial Neural Networks

1. Implementation of neural network using first principle:

Input Parameters:-

Hidden layer List : A list of number of neurons in each hidden layer

Batch Size: Mini batch size

Training Features : X

Train Labels : Y

Number of epochs

Activation Function : sigmoid or relu

1. hidden layer units= {5, 10, 15, 20, 25}

learning rate = 0.1.

mini-batch size =100 examples.

Stopping criteria= 1500 epochs

Error : 5.806288536781198

No of total values Match : 60000

number of correct match : 55909

Accuracy: 93.18166666666667

No of total values Match : 10000

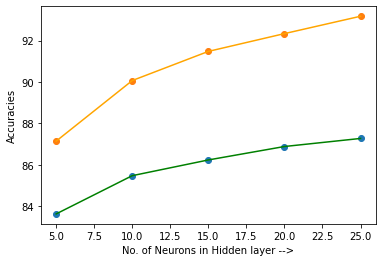
number of correct match : 8727

Accuracy: 87.27

Training Accuracies : [87.14, 90.07, 91.48, 92.34, 93.18]

Testing Accuracies : [83.62, 85.47, 86.23, 86.88, 87.27]

Time Taken : [1560, 1956, 1457, 1287, 1320]



Observation: The increase in the number of neurons in the hidden layer, increases the time to train the model , accuracy on both test and train sets.

1. Adaptive learning rate

hidden layer units= {5, 10, 15, 20, 25}

learning rate = 0.1.

mini-batch size =100 examples

stopping criteria= 1500 epochs

Error : 6.742214916764054

No of total values Match : 60000

number of correct match : 54053

Accuracy: 90.08833333333334

No of total values Match : 10000

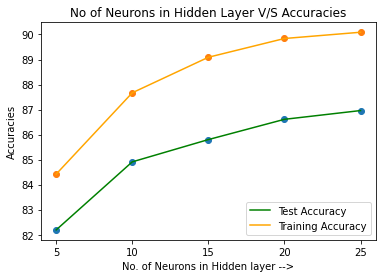
number of correct match : 8696

Accuracy: 86.96

Training Accuracies : [84.42, 87.67, 89.09, 89.84, 90.09]

Testing Accuracies : [82.19, 84.91, 85.8, 86.61, 86.96]

Time Taken : [992, 1125, 1116, 1306, 1312]



Observation: Accuracies have been reduced slightly in comparison to fixed learning rate. The time taken is more to train the network. From this graph we can see that the accuracies improve as the number of neurons in the hidden layer increases.

1. Relu acitvation function and how it is different from Sigmoid function:
2. Sigmoid

Hidden Layer = [100,100]

Activation: relu

Solver = sgd

Batch Size = 100

Epochs = 1500

Learning Type = Adaptive

Training accuracy= 92.45%

Test accuracy= 88.35%

Error : 29.564096572449888

No of total values Match : 60000

number of correct match : 28428

Accuracy: 47.38

No of total values Match : 10000

number of correct match : 4586

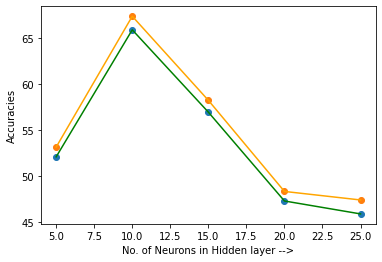
Accuracy: 45.86

Training Accuracies : [53.14, 67.4, 58.24, 48.31, 47.38]

Testing Accuracies : [52.08, 65.92, 56.95, 47.27, 45.86]

Time Taken : [977, 1068, 989, 1200, 1346]

Obs: the relu function here with adaptive learning type very poor test and train accuracies.



We can see from the graph that the accuracies are decreasing as the number of neuron s in hidden layer increase.

Here what we witness is a dying relu, among the limitations of ReLU is the case where large weight updates can mean that the summed input to the activation function is always negative, regardless of the input to the network. This means that a node with this problem will forever output an activation value of 0.0. This is referred to as a “dying ReLU“

* Relu activation function:

Hidden Layer = [100,100]

Activation: sigmoid

Solver = sgd

Batch Size = 100

Epochs = 1500

Learning Type = Adaptive

Error : 6.026265831244133

No of total values Match : 60000

number of correct match : 55475

Accuracy: 92.45833333333333

No of total values Match : 10000

number of correct match : 8835

Accuracy: 88.35

Where as the model ran using the sigmoid function gave better test and train accuracies.

1. Varying the number of hidden layers and the neurons per layer :

Hidden layers= [50,50,50,50,50]

* Sigmoid function:

Error : 6.526810489956394

No of total values Match : 60000

number of correct match : 54566

Accuracy: 90.94333333333333

No of total values Match : 10000

number of correct match : 8767

Accuracy: 87.67

Training Accuracies : [90.95, 90.89, 90.86, 91.06, 90.94]

Testing Accuracies : [87.75, 88.02, 87.84, 87.75, 87.67]

Time Taken : [1751, 1618, 1445, 1555, 1569]

Here the accuracies of the sigmoid activation function yield better results.

* Relu function:

Error : 32.154925692782925

No of total values Match : 60000

number of correct match : 23803

Accuracy: 39.67166666666667

No of total values Match : 10000

number of correct match : 3932

Accuracy: 39.32

Training Accuracies : [49.05, 58.37, 49.12, 67.4, 39.67]

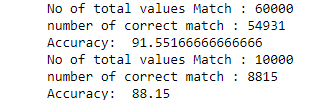
Testing Accuracies : [48.0, 57.06, 48.38, 65.62, 39.32]

Time Taken : [1699, 2832, 2551, 2015, 2303]

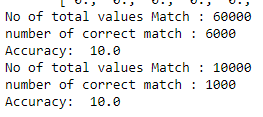
Overall the relu activation function doesn’t yield good results for our data.

F) Binary Cross-entroy

* Sigmoid



* Relu



Here also after applying binary cross entropy to our same network architecture yield better results for the sigmoid function as compared to relu activation function.

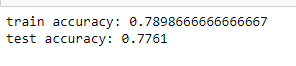
G) Multi-Layer Perceptron Model:

* For sigmoid function:



The network I build yields better accuracies for Sigmoid activation function in both fixed and adaptive learning types.

* For relu function :



After directly applying the MLPClassifier to our dataset we retrieve almost same accuracies for both realu and sigmoid function.

Also, the the network I build yields poor accuracies for Relu activation function in adaptive learning type.