Introduction to Neural Networks
Homework #2

機械所
張元睿
N16054629

October 21, 2016

The derivations/development of the learning algorithms

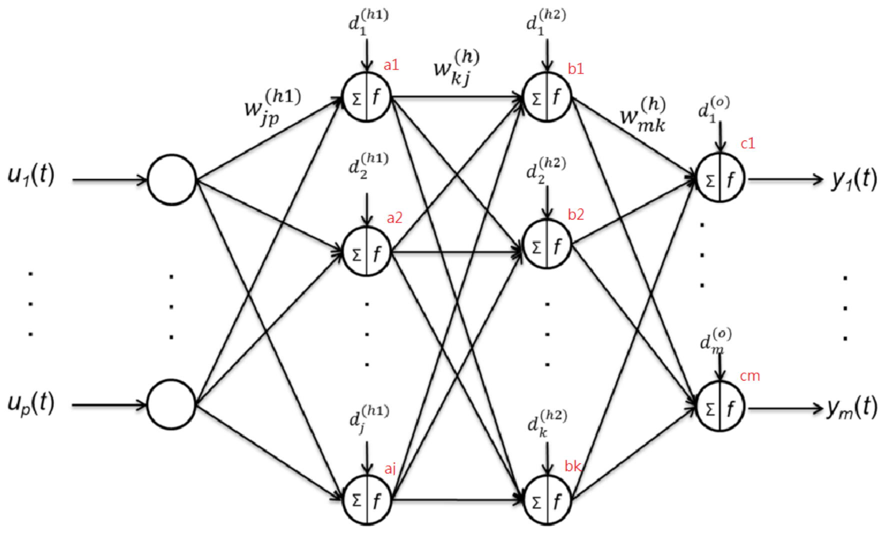


Fig. 1. Structure of three-layer feedforward neural network.

1. Forward path:

$$a_j = \sum_{\alpha=1\sim j,\ \beta=1\sim p} w_{\alpha\beta}^{(h1)} \times u_\beta(t) + d_\alpha^{(h1)}$$

$$Y_{a_j} = f(a_j)$$

$$b_k = \sum_{\alpha=1\sim k,\ \beta=1\sim j} w_{\alpha\beta}^{(h2)} \times Y_{a_\beta} + d_\alpha^{(h2)}$$

$$Y_{b_k} = g(b_k)$$

$$y_m(t) = c_m = \sum_{\alpha=1\sim m,\ \beta=1\sim k} w_{\alpha\beta}^{(h3)} \times y_{b_\beta} + d_\alpha^{(o)}$$

$$E_m(t) = \tfrac{1}{2} e_m^2(t)$$

$$e_m(t) = d_m(t) - y_m(t)$$

for part1 $d_m(t)$ is composed of 1 and 0

2. Backward propagation

   (a) Update rule for the weights of the output neurons:

$$
\begin{aligned}
w_{mk}^{(h_3)}(t+1) &= w_{mk}^{(h_3)}(t) + \Delta w_{mk}(t) \\
&= w_{mk}^{(h_3)}(t) - \eta \frac{\partial E_m(t)}{\partial w_{mk}^{(h_3)}(t)} \\
&= w_{mk}^{(h_3)}(t) - \eta \frac{\partial E_m(t)}{\partial e_m(t)} \frac{\partial e_m(t)}{\partial y_m(t)} \frac{\partial y_m(t)}{\partial c_m(t)} \frac{\partial c_m(t)}{\partial w_{mk}^{(h_3)}(t)} \\
&= w_{mk}^{(h_3)}(t) - \eta (d_m(t) - y_m(t))(-1)(1)(Y_{b_k}(t)) \\
&= w_{mk}^{(h_3)}(t) + \eta (d_m(t) - y_m(t))(Y_{b_k}(t))
\end{aligned}
$$

   (b) Update rule for the biases of the output neurons:

$$
\begin{aligned}
d_m^{(o)}(t+1) &= d_m^{(o)}(t) + \Delta d_m(t) \\
&= d_m^{(o)}(t) - \eta \frac{\partial E_m(t)}{\partial d_m^{(o)}(t)} \\
&= d_m^{(o)}(t) - \eta \frac{\partial E_m(t)}{\partial e_j(t)} \frac{\partial e_m(t)}{\partial y_m(t)} \frac{\partial y_m(t)}{\partial c_m(t)} \frac{\partial c_m(t)}{\partial d_m^{(o)}(t)} \\
&= d_m^{(o)}(t) - \eta (d_m(t) - y_m(t))(-1)(1)(1) \\
&= d_m^{(o)}(t) + \eta (d_m(t) - y_m(t))
\end{aligned}
$$

(c) Update rule for the weights of the second hidden neurons:

$$\begin{aligned}
w_{kj}^{(h_2)}(t+1) &= w_{kj}^{(h_2)}(t) + \Delta w_{kj}(t) \\
&= w_{kj}^{(h_2)}(t) - \eta \frac{\partial E_m(t)}{\partial w_{kj}^{(h_2)}(t)} \\
&= w_{kj}^{(h_2)}(t) - \eta \frac{\partial E_m(t)}{\partial e_m(t)} \frac{\partial e_m(t)}{\partial y_m(t)} \frac{\partial y_m(t)}{\partial c_m(t)} \frac{\partial c_m(t)}{\partial Y_{b_k}(t)} \frac{\partial Y_{b_k}(t)}{\partial b_k(t)} \frac{\partial b_k(t)}{\partial w_{kj}^{(h_2)}(t)} \\
&= w_{kj}^{(h_2)}(t) - \eta \sum_m (d_m(t) - y_m(t))(-1)(1)(w_{mk}^{(h_3)}(t)) g'(b_k(t))(Y_{a_j}(t)) \\
&= w_{kj}^{(h_2)}(t) + \eta \sum_m (d_m(t) - y_m(t))(w_{mk}^{(h_3)}(t)) g'(b_k(t))(Y_{a_j}(t))
\end{aligned}$$

(d) Update rule for the biases of the second hidden neurons:

$$\begin{aligned}
d_k^{(h_2)}(t+1) &= d_k^{(h_2)}(t) + \Delta d_k(t) \\
&= d_k^{(h_2)}(t) - \eta \frac{\partial E_m(t)}{\partial d_k^{(h_2)}(t)} \\
&= d_k^{(h_2)}(t) - \eta \frac{\partial E_m(t)}{\partial e_j(t)} \frac{\partial e_m(t)}{\partial y_m(t)} \frac{\partial y_m(t)}{\partial c_m(t)} \frac{\partial c_m(t)}{\partial Y_{b_k}(t)} \frac{\partial Y_{b_k}(t)}{\partial b_k(t)} \frac{\partial b_k(t)}{\partial d_j^{(h_1)}(t)} \\
&= d_k^{(h_2)}(t) - \eta \sum_m (d_m(t) - y_m(t))(-1)(1)(w_{mk}^{(h_3)}(t)) g'(b_k(t))(1) \\
&= d_k^{(h_2)}(t) + \eta \sum_m (d_m(t) - y_m(t))(w_{mk}^{(h_3)}(t)) g'(b_k(t))
\end{aligned}$$

(e) Update rule for the weights of the first hidden neurons:

$$
\begin{aligned}
w_{jp}^{(h_1)}(t+1) &= w_{jp}^{(h_1)}(t) + \Delta w_{jp}(t) \\
&= w_{jp}^{(h_1)}(t) - \eta \frac{\partial E_m(t)}{\partial w_{jp}^{(h_1)}(t)} \\
&= w_{jp}^{(h_1)}(t) - \eta \frac{\partial E_m(t)}{\partial e_m(t)} \frac{\partial e_m(t)}{\partial y_m(t)} \frac{\partial y_m(t)}{\partial c_m(t)} \frac{\partial c_m(t)}{\partial Y_{b_k}(t)} \frac{\partial Y_{b_k}(t)}{\partial b_k(t)} \frac{\partial b_k(t)}{\partial Y_{aj}(t)} \\
&\qquad \frac{\partial Y_{aj}(t)}{\partial a_j(t)} \frac{\partial a_j(t)}{\partial w_{jp}^{(h_1)}(t)} \\
&= w_{jp}^{(h_1)}(t) - \eta \sum_k \sum_m (d_m(t) - y_m(t))(-1)(1)(w_{mk}^{(h_3)}(t)) g'(b_k(t)) \\
&\qquad (w_{kj}^{(h_2)}(t)) f'(a_j(t))(u_p(t)) \\
&= w_{jp}^{(h_1)}(t) + \eta \sum_k \sum_m (d_m(t) - y_m(t))(w_{mk}^{(h_3)}(t)) g'(b_k(t))(w_{kj}^{(h_2)}(t)) \\
&\qquad f'(a_j(t))(u_p(t))
\end{aligned}
$$

(f) Update rule for the biases of the first hidden neurons:

$$d_j^{(h_1)}(t+1) = d_j^{(h_1)}(t) + \Delta d_j(t)$$

$$= d_j^{(h_1)}(t) - \eta \frac{\partial E_m(t)}{\partial d_j^{(h_1)}(t)}$$

$$= d_j^{(h_1)}(t) - \eta \frac{\partial E_m(t)}{\partial e_j(t)} \frac{\partial e_m(t)}{\partial y_m(t)} \frac{\partial y_m(t)}{\partial c_m(t)} \frac{\partial c_m(t)}{\partial Y_{b_k}(t)} \frac{\partial Y_{b_k}(t)}{\partial b_k(t)} \frac{\partial b_k(t)}{\partial Y_{a_j}(t)}$$

$$\frac{\partial Y_{aj}(t)}{\partial a_j(t)} \frac{\partial a_j(t)}{\partial d_j^{(h_1)}(t)}$$

$$= d_j^{(h_1)}(t) - \eta \sum_k \sum_m (d_m(t) - y_m(t))(-1)(1)(w_{mk}^{(h_3)}(t))g'(b_k(t))$$

$$(w_{kj}^{(h_2)}(t))f'(a_j(t))(1)$$

$$= d_j^{(h_1)}(t) + \eta \sum_k \sum_m (d_m(t) - y_m(t))(w_{mk}^{(h_3)}(t))g'(b_k(t))(w_{kj}^{(h_2)}(t))f'(a_j(t))$$

In this homework, we will use

i. hyperbolic tangent functions for all the hidden layers

ii. sigmoid functions for all the hidden layers

iii. hyperbolic tangent functions for the first hidden layer and sigmoid functions for the second layer

as the activation function $f()$ and $g()$ to evaluate the network performance.

Part1: classification

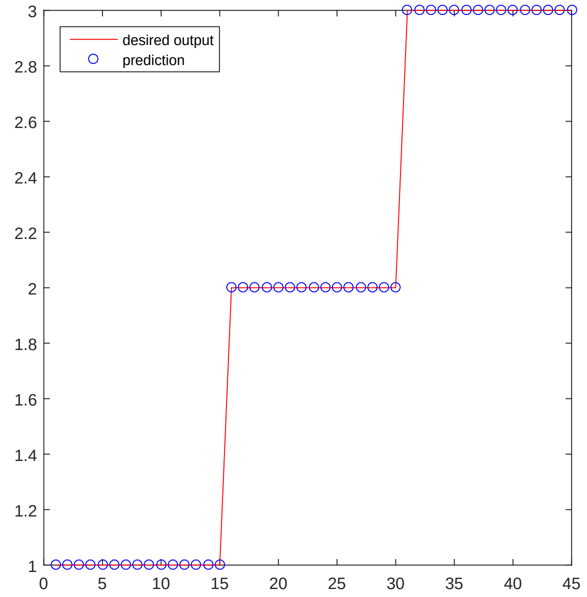
1. Iris

   (a) topologies (structures) of the networks:
   input layer node: 4
   first hidden layer node: 2
   second hidden layer node: 3
   output layer node: 3
   all layers are fully connected

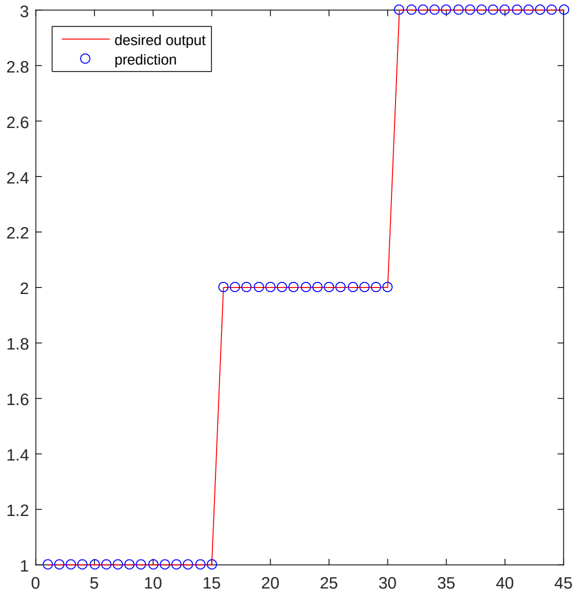
   (b) best three results out of 10 trials using different initializations:

      i. hyperbolic tangent functions for all the hidden layers:
      learning rate = 0.1
      epoch = 20

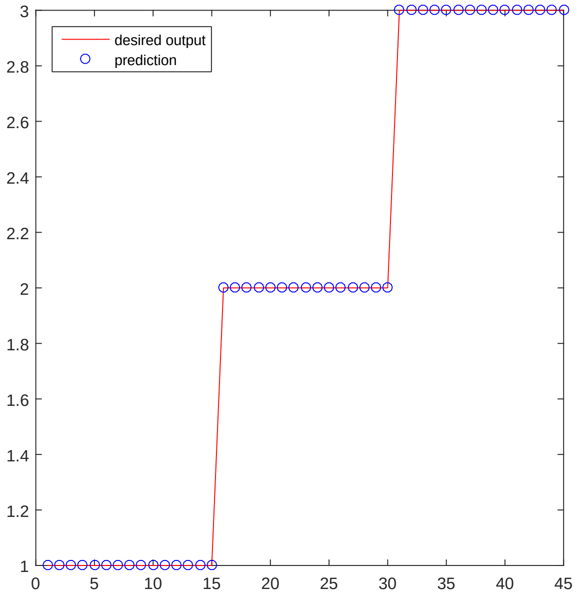
         A. accuracy = 100%



B. accuracy = 100%

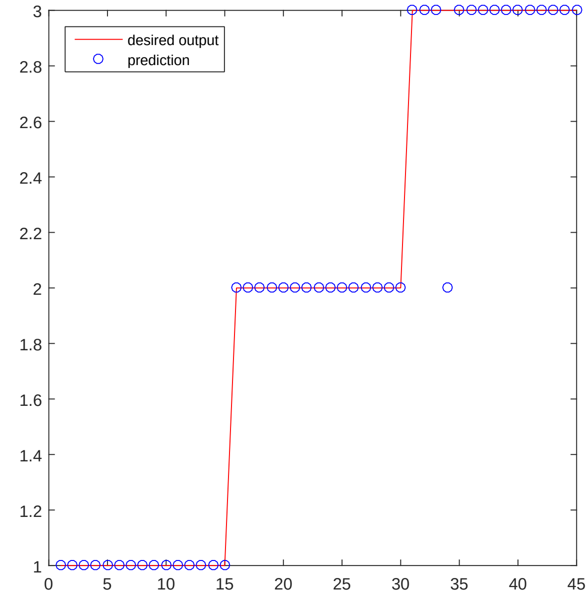


C. accuracy = 100%

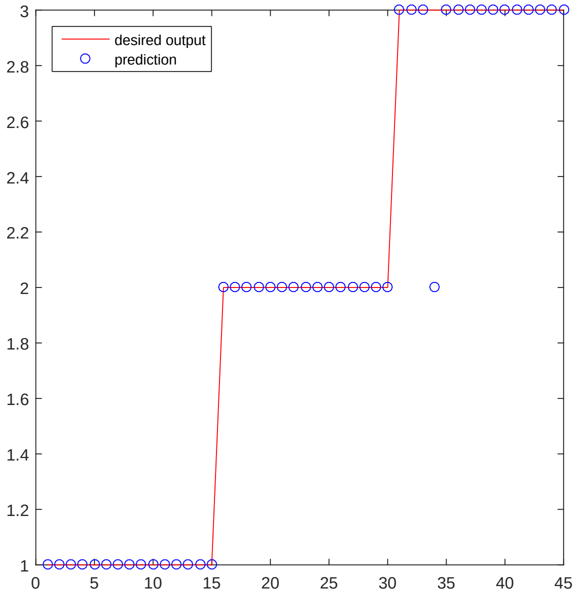


ii. sigmoid functions for all the hidden layers:
learning rate = 0.1
epoch = 50

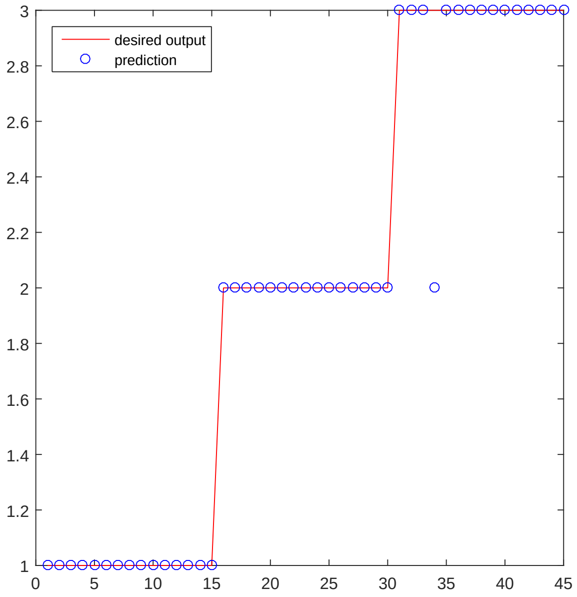
A. accuracy = 97.8%



B. accuracy = 97.8%

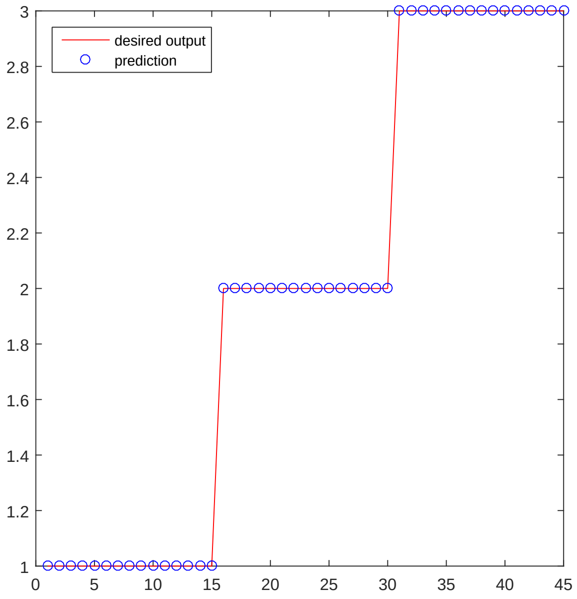


C. accuracy = 97.8%

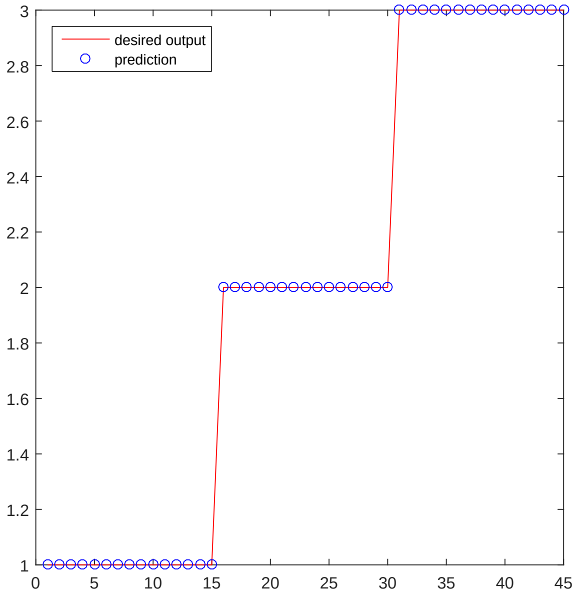


iii. hyperbolic tangent functions for the first hidden layer and sigmoid functions for the second layer:
learning rate = 0.1
epoch = 20

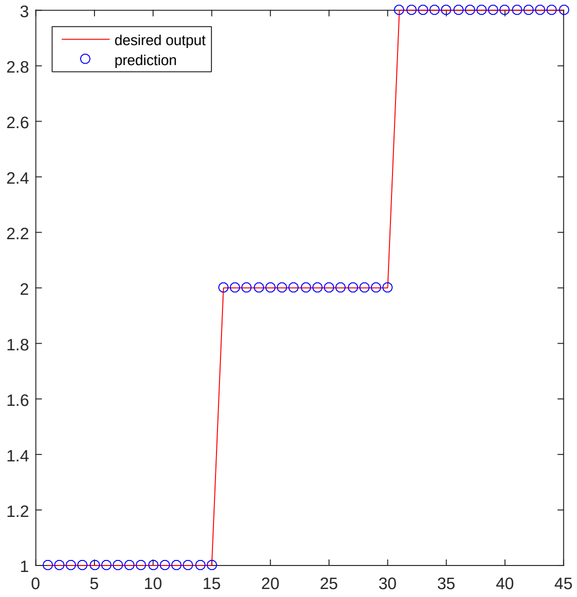
A. accuracy = 100%



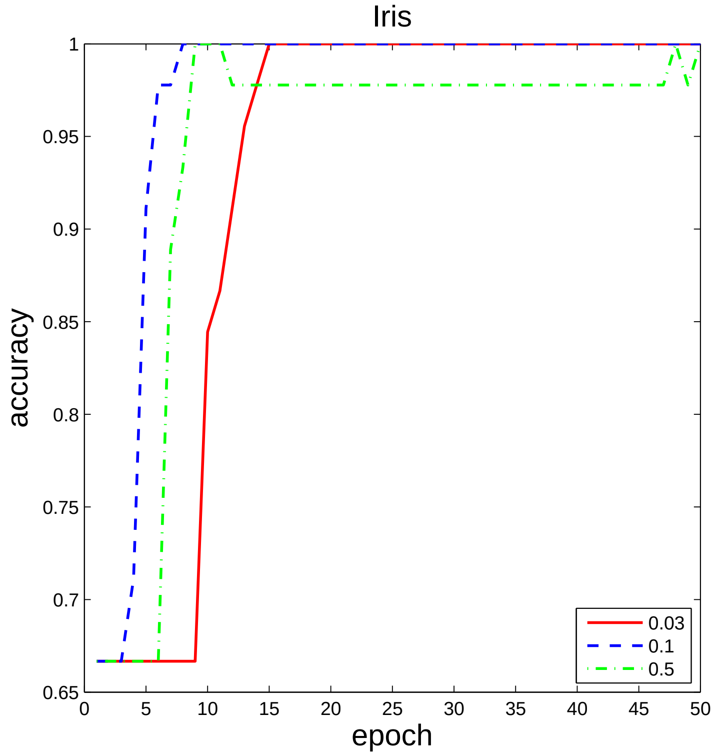
B. accuracy = 100%



C. accuracy = 100%



(c) For the random weight assignment, use the same initial weights that you obtain the best classification result to compare the learning curves with different learning rates:



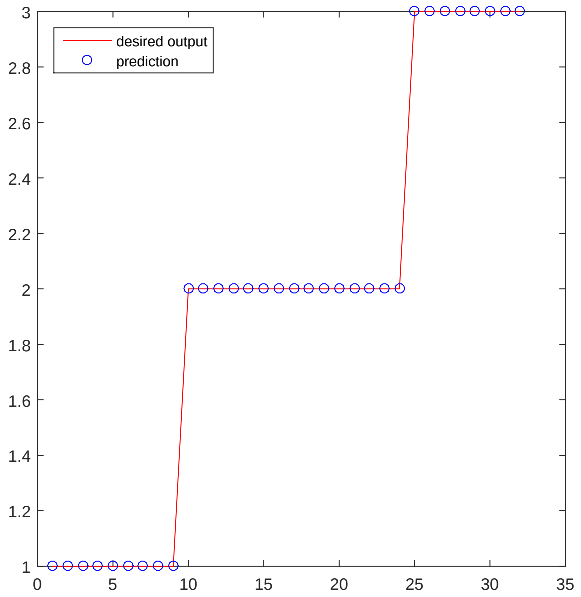
2. Wine

   (a) topologies (structures) of the networks:
   input layer node: 13
   first hidden layer node: 4
   second hidden layer node: 2
   output layer node: 3
   all layers are fully connected

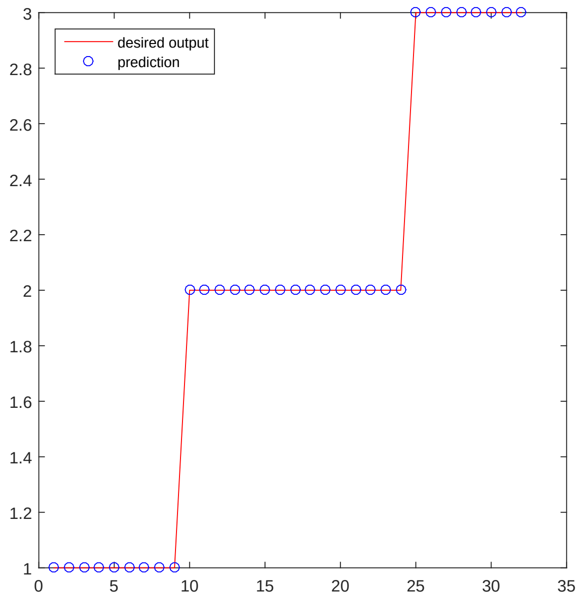
   (b) best three results out of 10 trials using different initializations:

      i. hyperbolic tangent functions for all the hidden layers:
      learning rate = 0.1
      epoch = 30

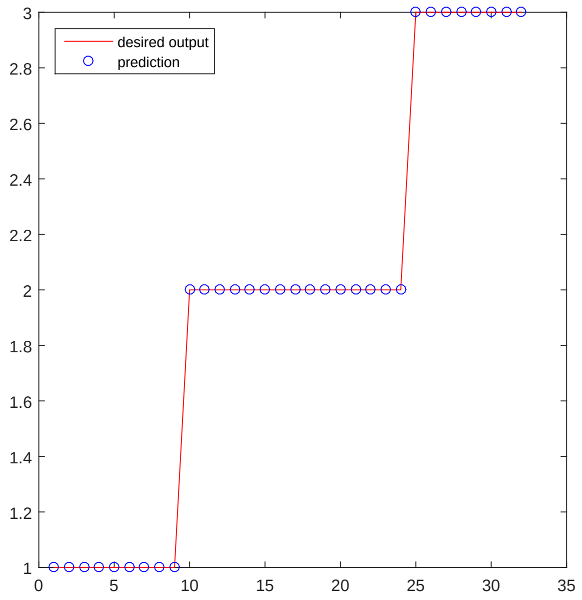
         A. accuracy = 100%



B. accuracy = 100%

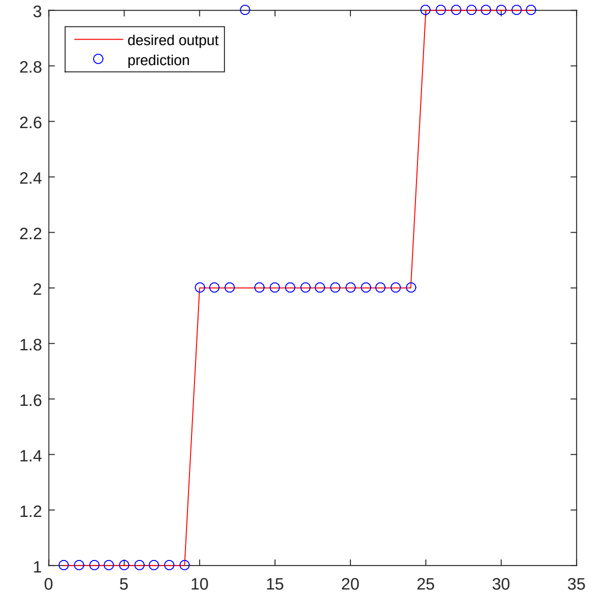


C. accuracy = 100%

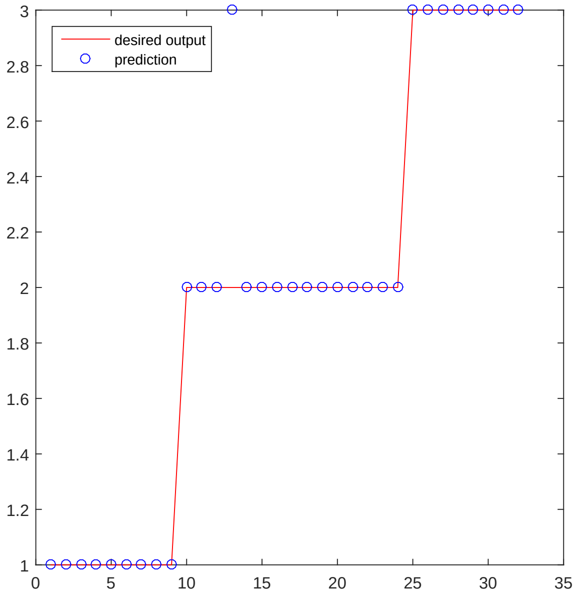


ii. sigmoid functions for all the hidden layers:
learning rate = 0.1
epoch = 30

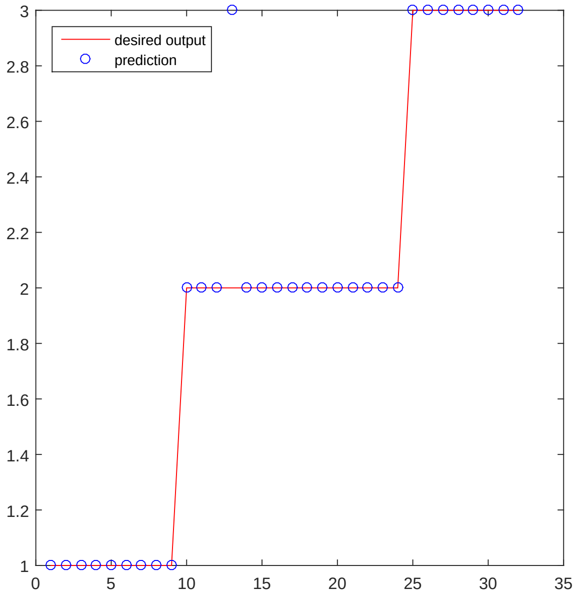
A. accuracy = 96.9%



B. accuracy = 96.9%

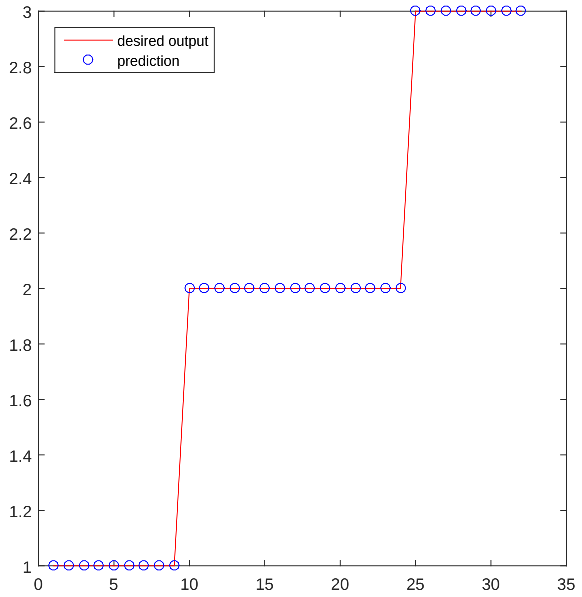


C. accuracy = 96.9%

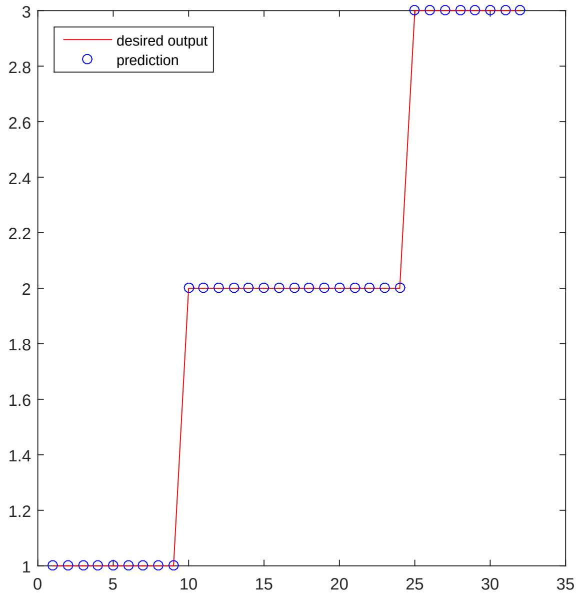


iii. hyperbolic tangent functions for the first hidden layer and sigmoid functions for the second layer:
learning rate $= 0.1$
epoch $= 30$

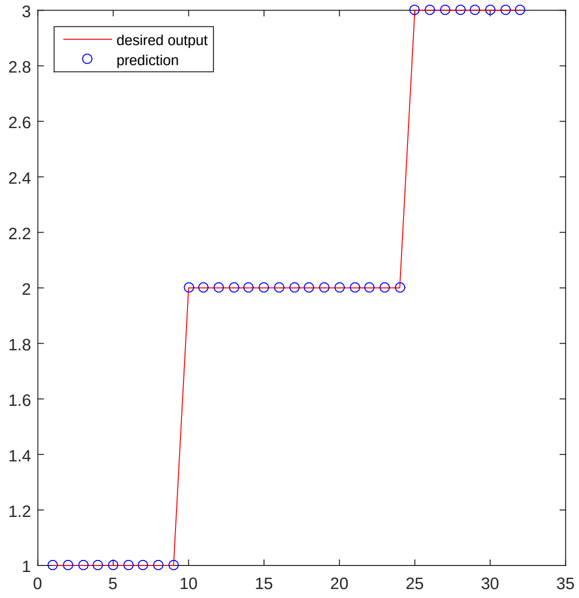
A. accuracy $= 100\%$



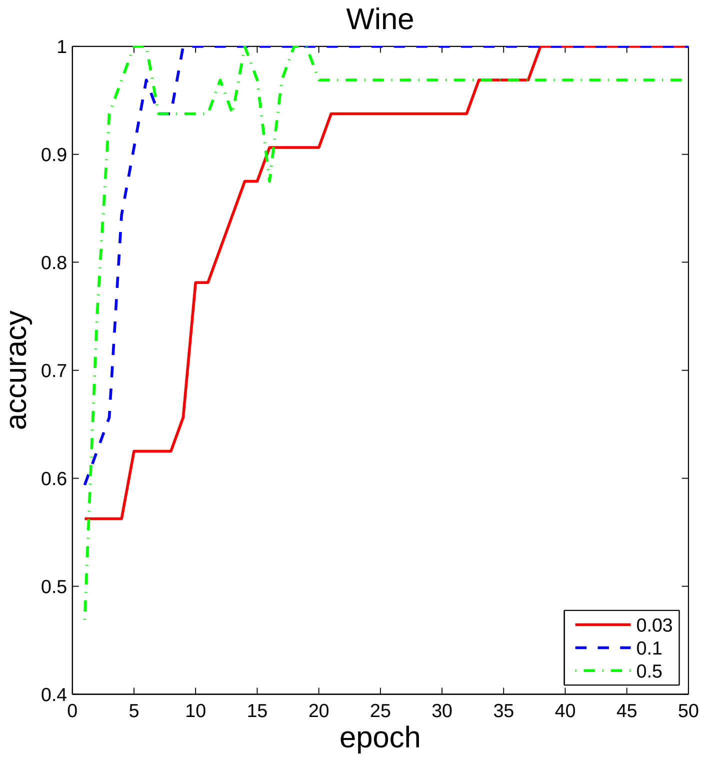
B. accuracy = 100%



C. accuracy = 100%



(c) For the random weight assignment, use the same initial weights that you obtain the best classification result to compare the learning curves with different learning rates:



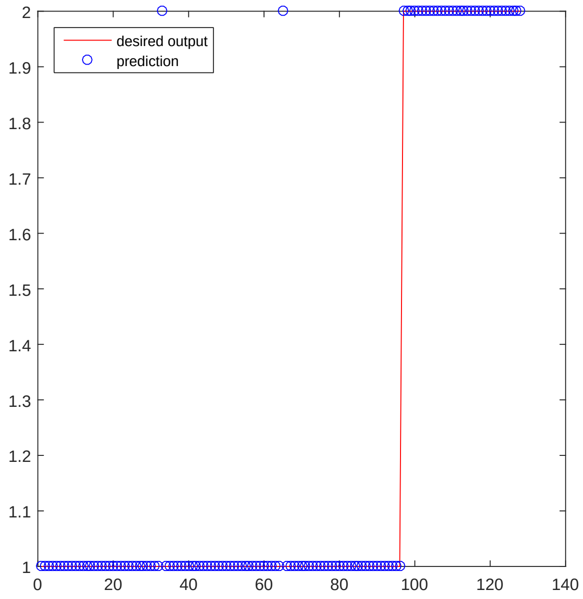
3. Breast Cancer Wisconsin

   (a) topologies (structures) of the networks:
   input layer node: 9
   first hidden layer node: 3
   second hidden layer node: 3
   output layer node: 2
   all layers are fully connected

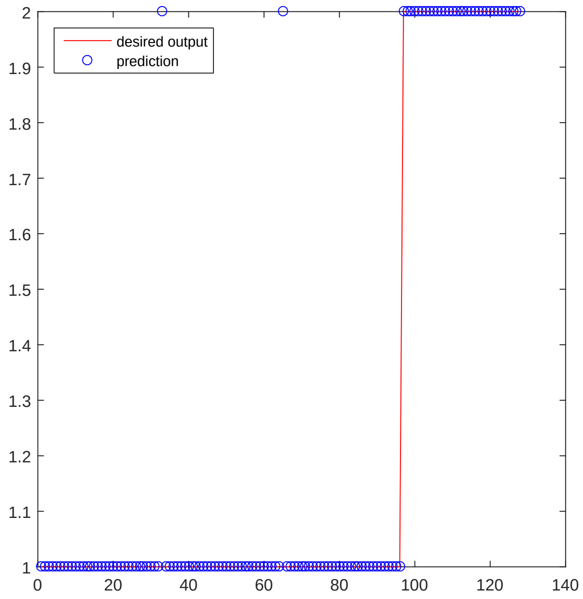
   (b) best three results out of 10 trials using different initializations:

      i. hyperbolic tangent functions for all the hidden layers:
      learning rate = 0.1
      epoch = 15

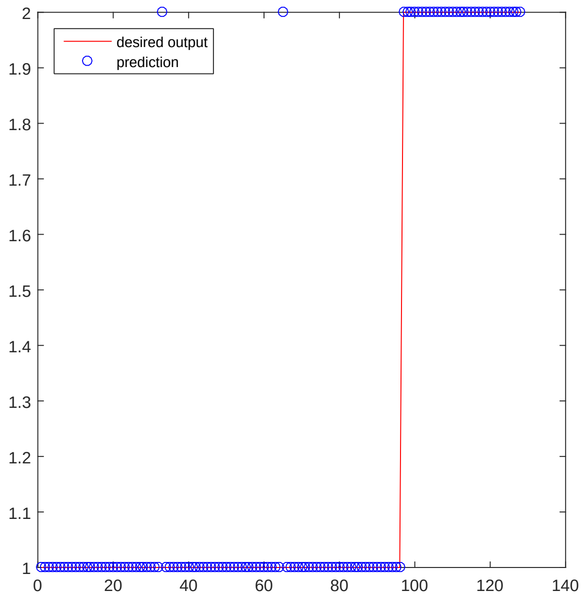
         A. accuracy = 98.4%



B. accuracy = 98.4%

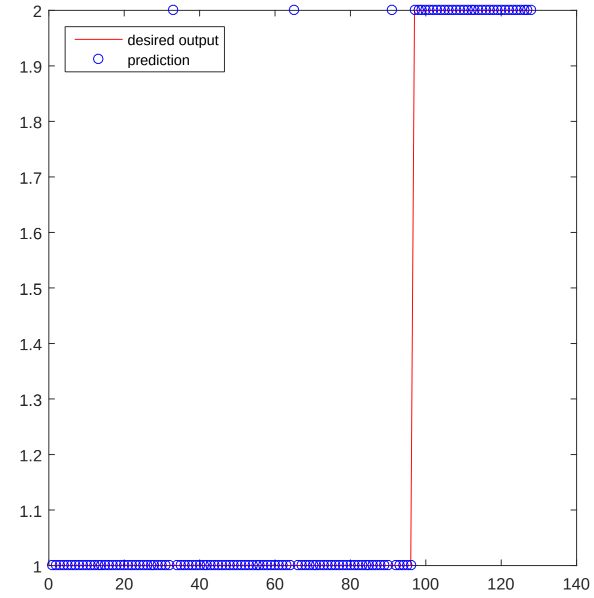


C. accuracy = 98.4%

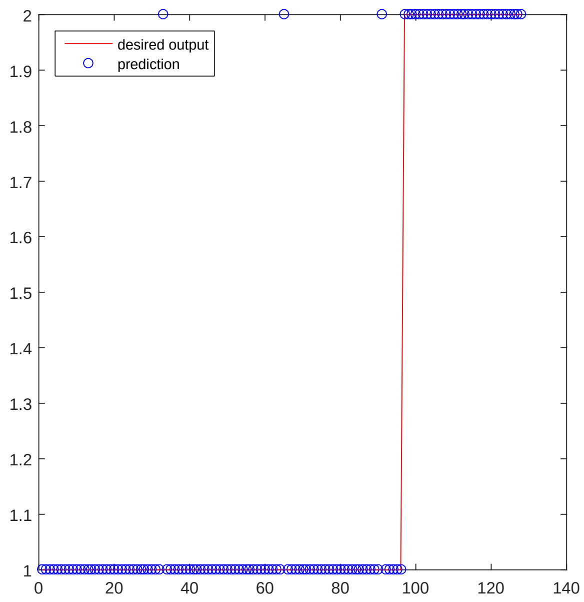


ii. sigmoid functions for all the hidden layers:
learning rate = 0.1
epoch = 20

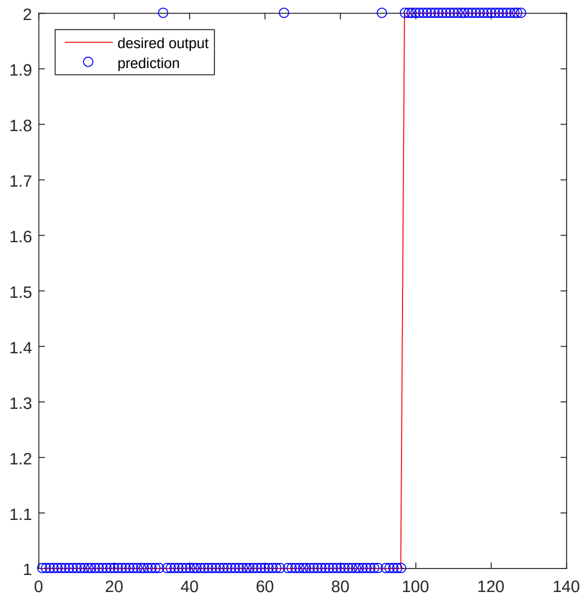
A. accuracy = 97.7%



B. accuracy = 97.7%

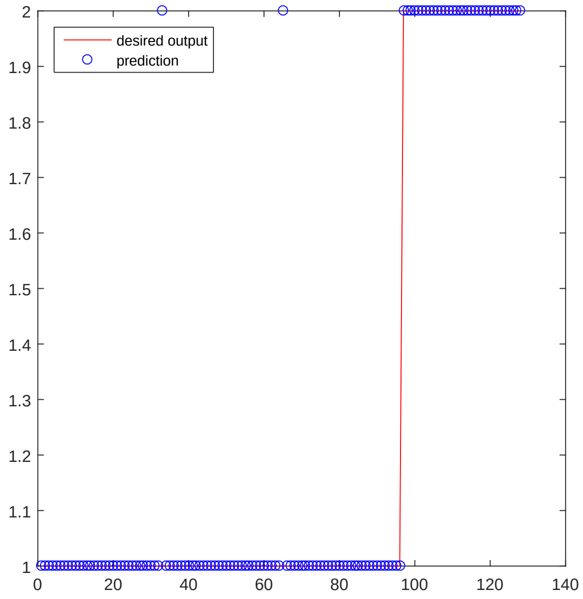


C. accuracy = 97.7%

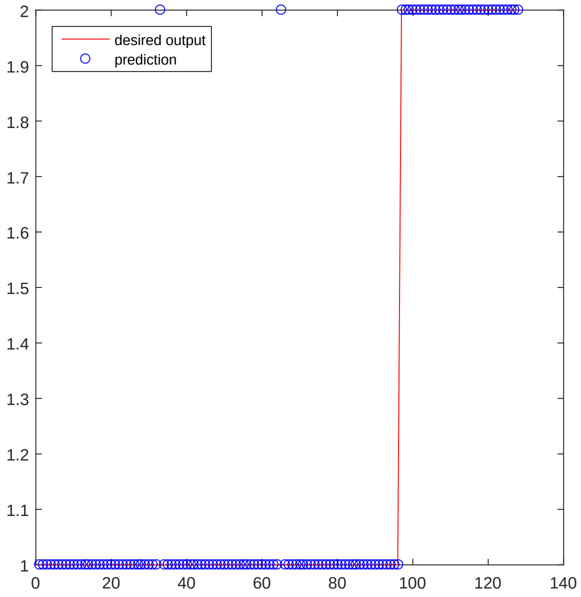


iii. hyperbolic tangent functions for the first hidden layer and sigmoid functions for the second layer:
learning rate $= 0.1$
epoch $= 20$

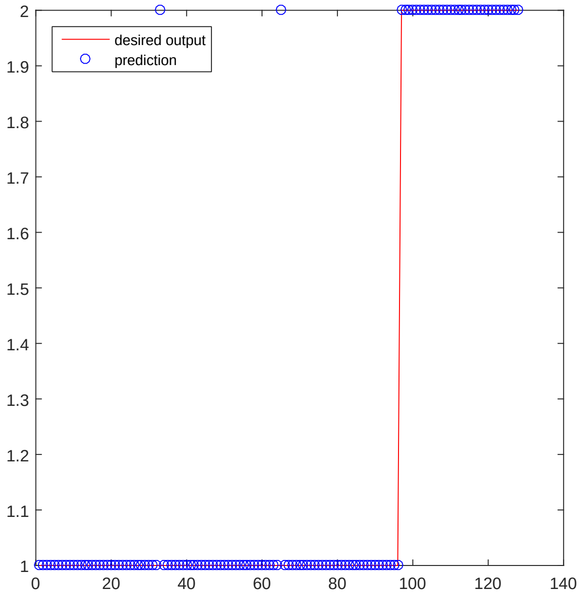
A. accuracy $= 98.4\%$



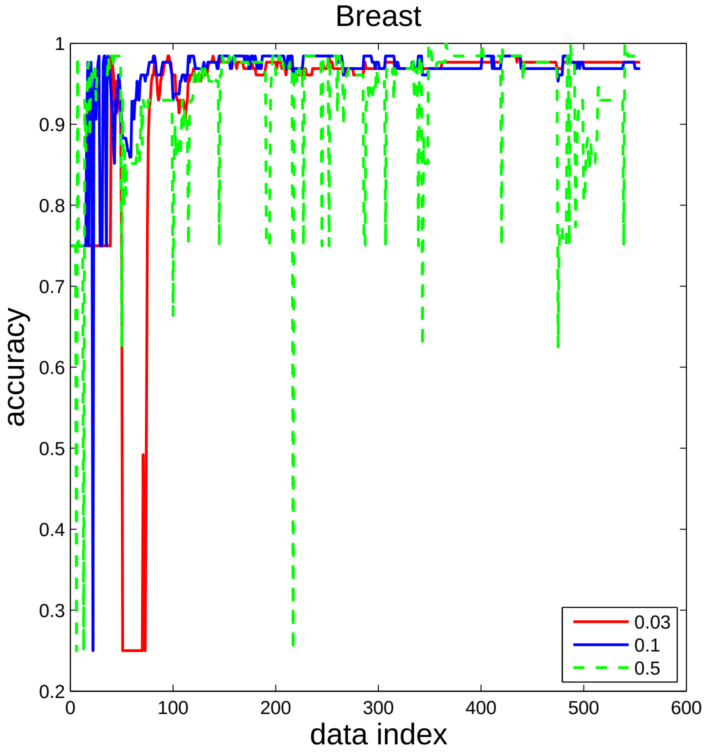
B. accuracy = 98.4%



C. accuracy = 98.4%



(c) For the random weight assignment, use the same initial weights that you obtain the best classification result to compare the learning curves with different learning rates:



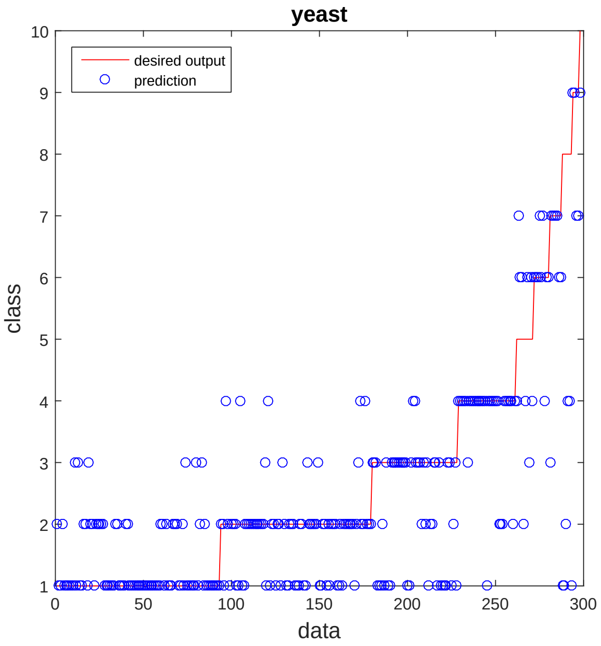
4. Yeast

   (a) topologies (structures) of the networks:
   input layer node: 8
   first hidden layer node: 5
   second hidden layer node: 7
   output layer node: 10
   all layers are fully connected

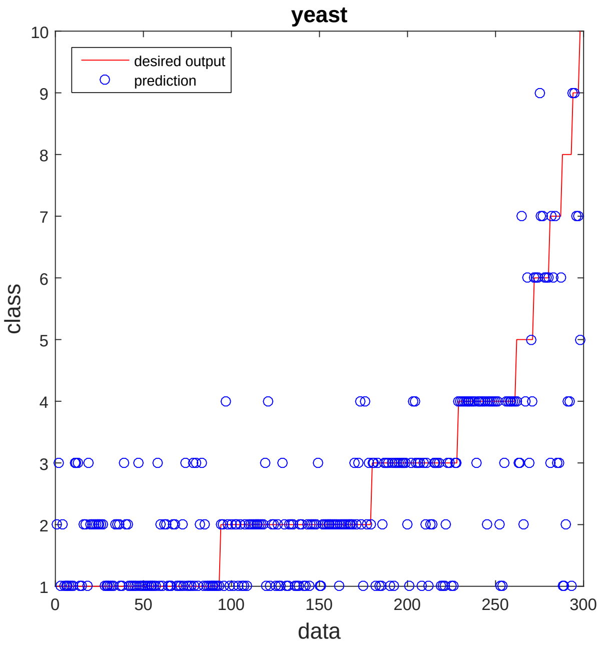
   (b) best three results out of 10 trials using different initializations:

      i. hyperbolic tangent functions for all the hidden layers:
      learning rate = 0.01
      epoch = 500

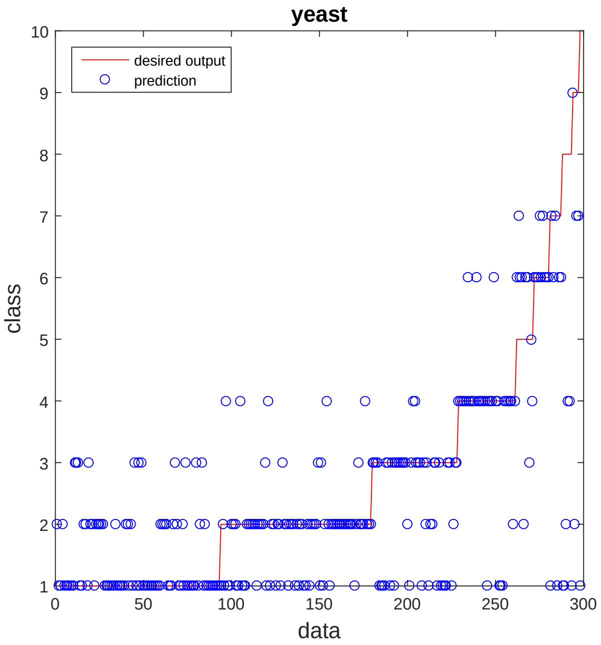
         A. accuracy = 59.7%



B. accuracy = 58.4%

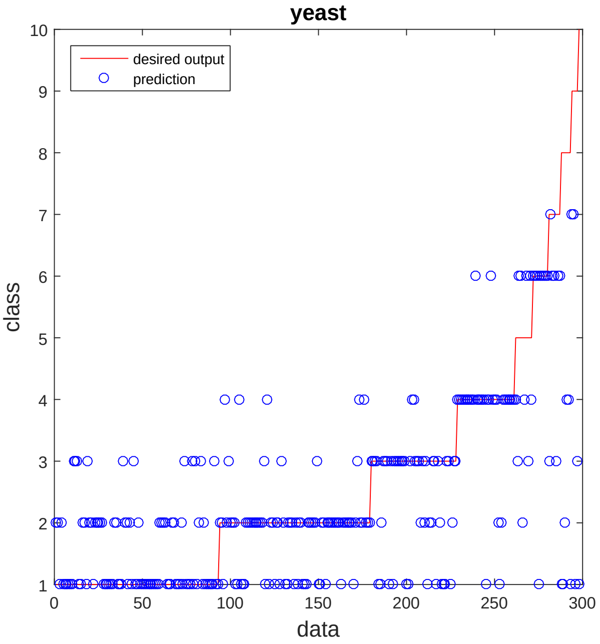


C. accuracy = 58.1%

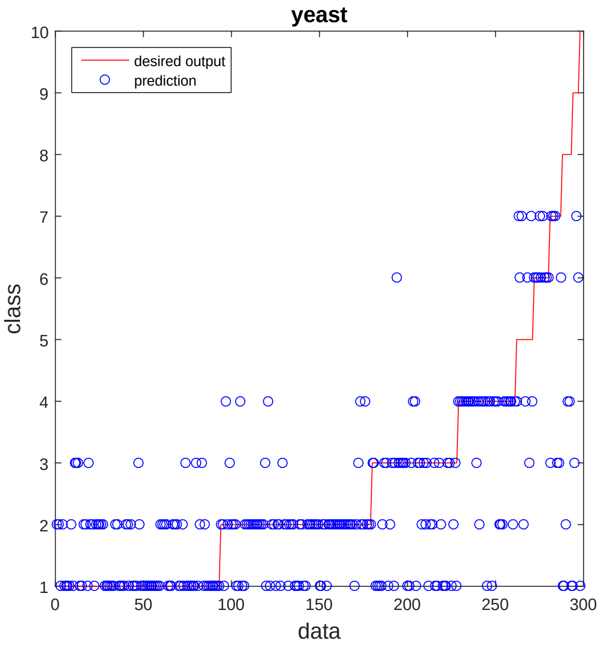


ii. sigmoid functions for all the hidden layers:
learning rate = 0.1
epoch = 500

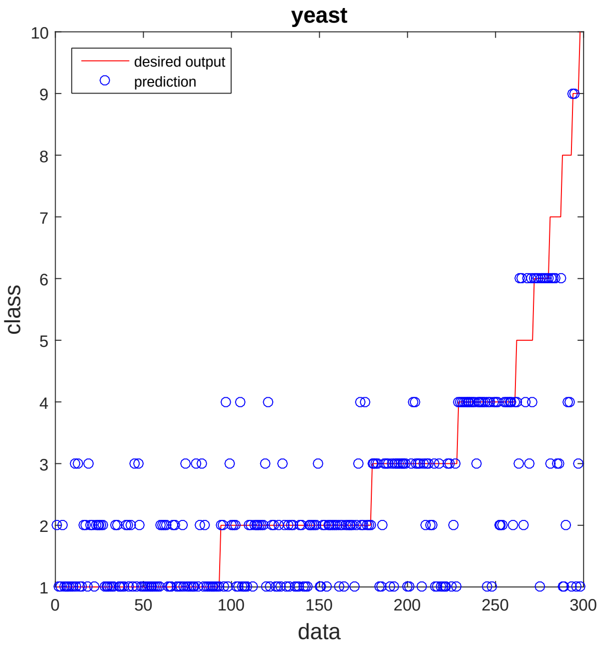
A. accuracy = 58.1%



B. accuracy = 57.1%

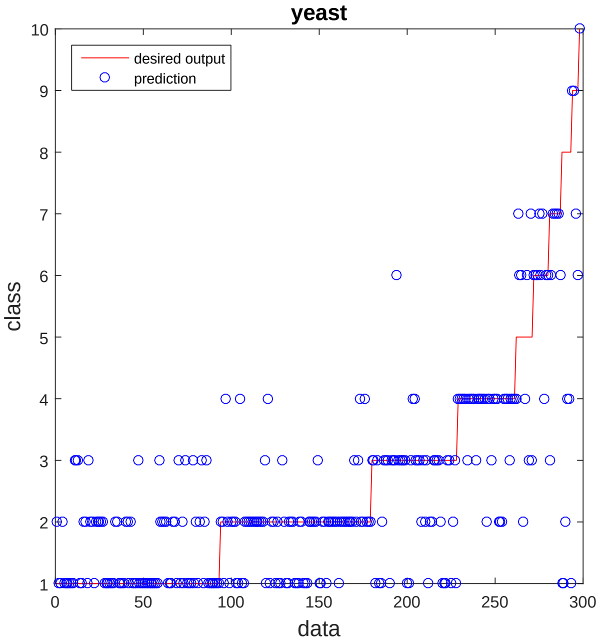


C. accuracy = 57.1%

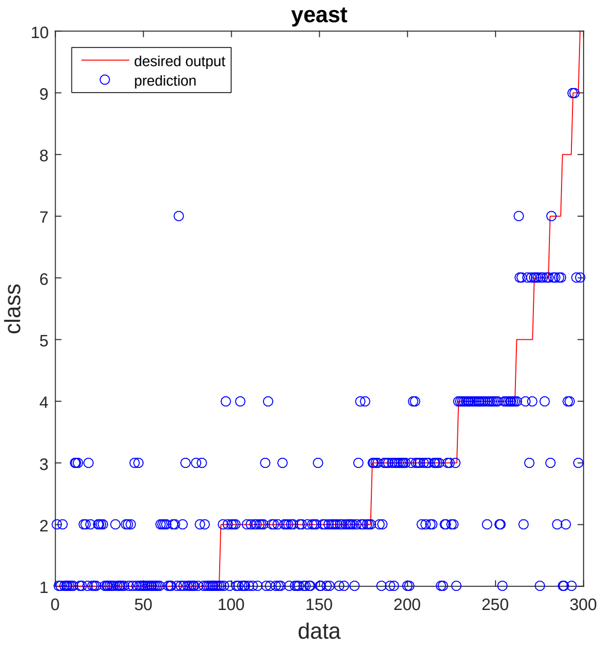


iii. hyperbolic tangent functions for the first hidden layer and sigmoid functions for the second layer:
learning rate = 0.1
epoch = 500

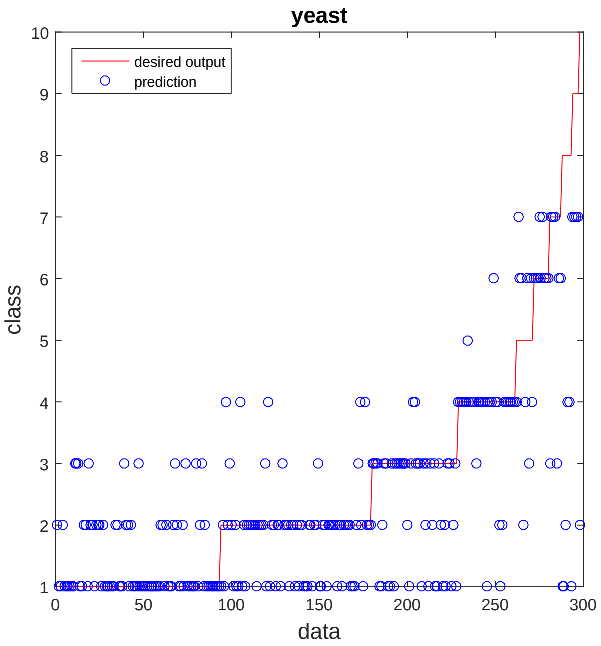
A. accuracy = 58.4%



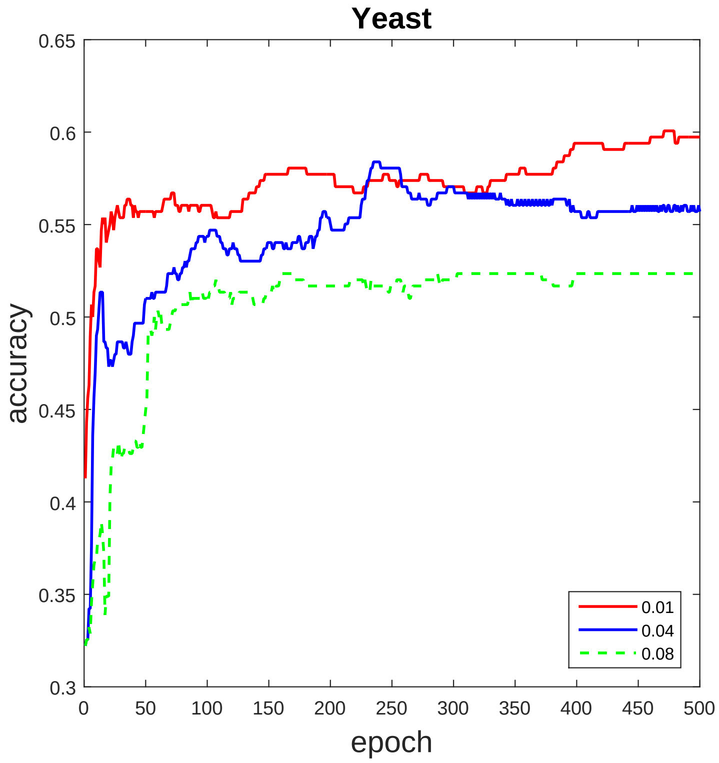
B. accuracy = 58.7%



C. accuracy = 57.1%



(c) For the random weight assignment, use the same initial weights that you obtain the best classification result to compare the learning curves with different learning rates:

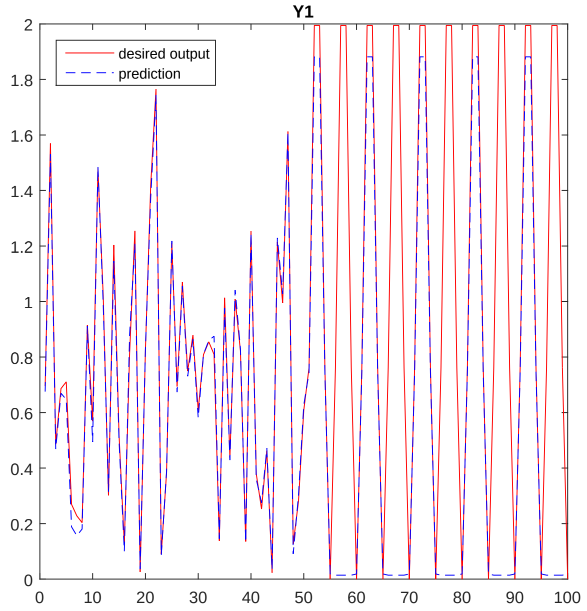


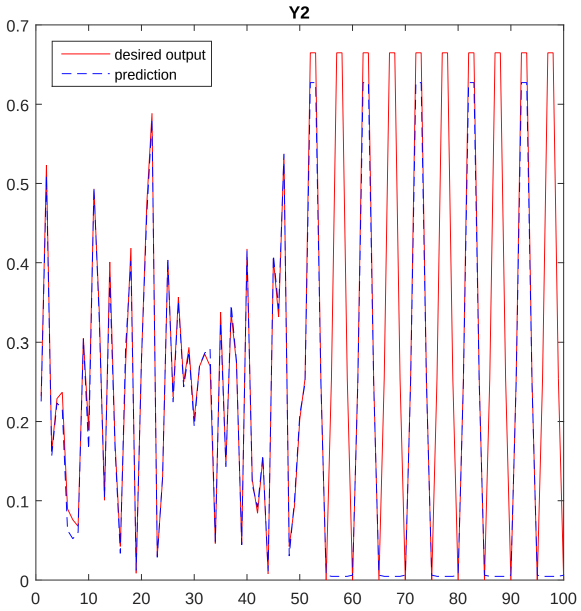
Part2: Function Approximation

1. topologies (structures) of the networks:
   input layer node: 2
   first hidden layer node: 2
   second hidden layer node: 3
   output layer node: 2
   all layers are fully connected

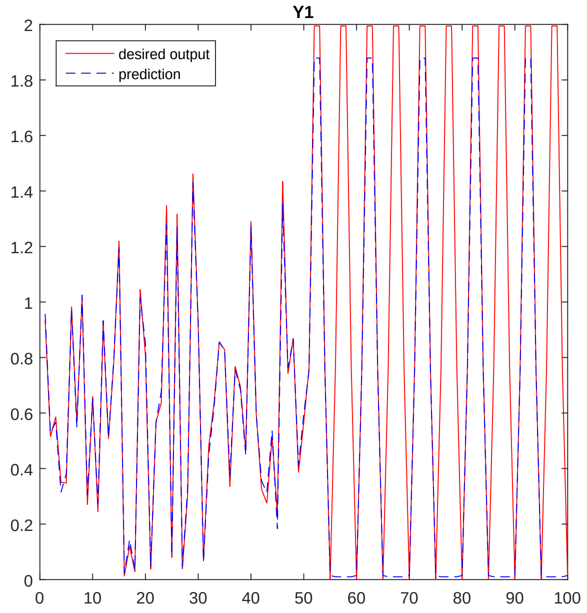
2. best three results out of 10 trials using different initializations:
   with learning rate =0.6, epoch =500

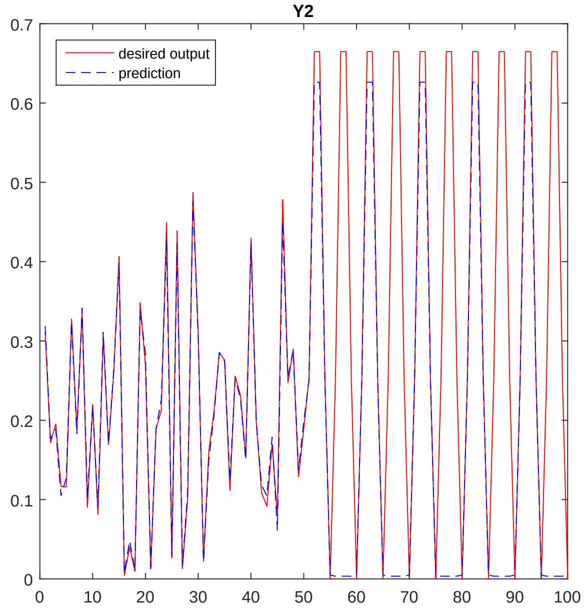
   (a) mean square errors $Y_1 = 0.004, Y_2 = 0.000448$



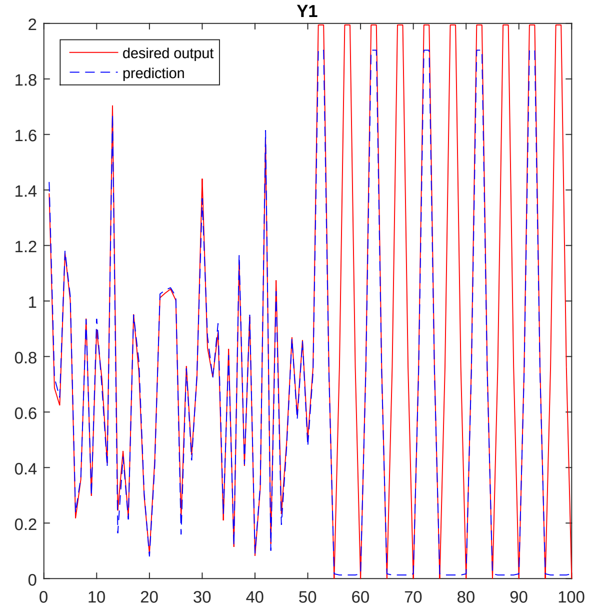

Y2
desired output
prediction
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
10
20
30
40
50
60
70
80
90
100

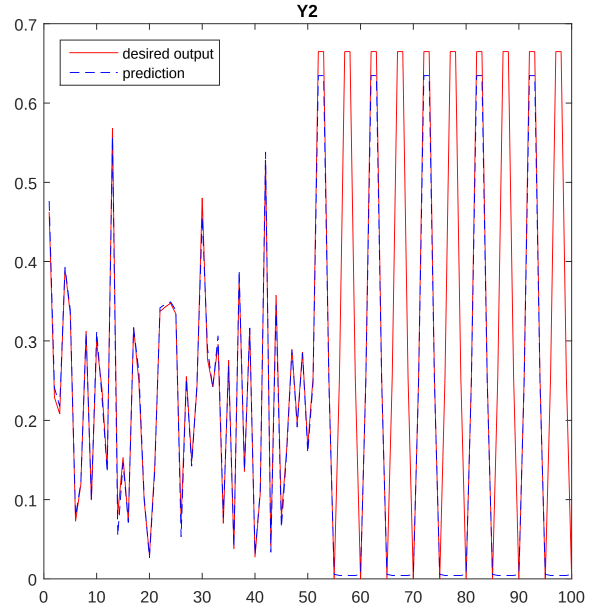
(b) mean square errors $Y_1 = 0.0041, Y_2 = 0.000452$





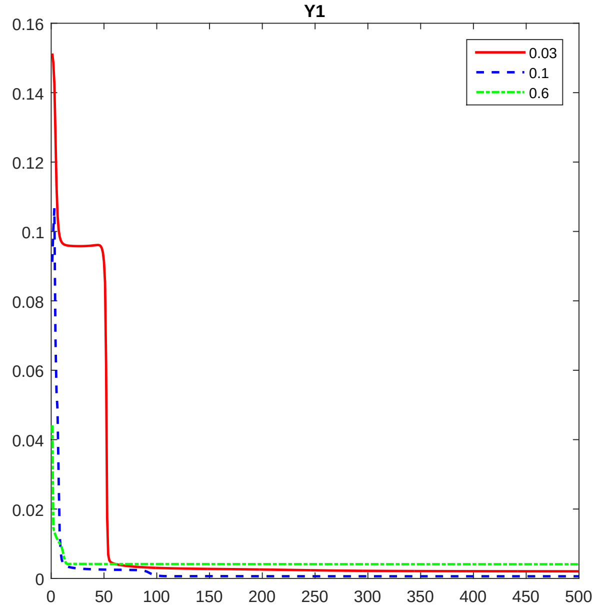
(c) mean square errors $Y_1 = 0.0041, Y_2 = 0.000454$

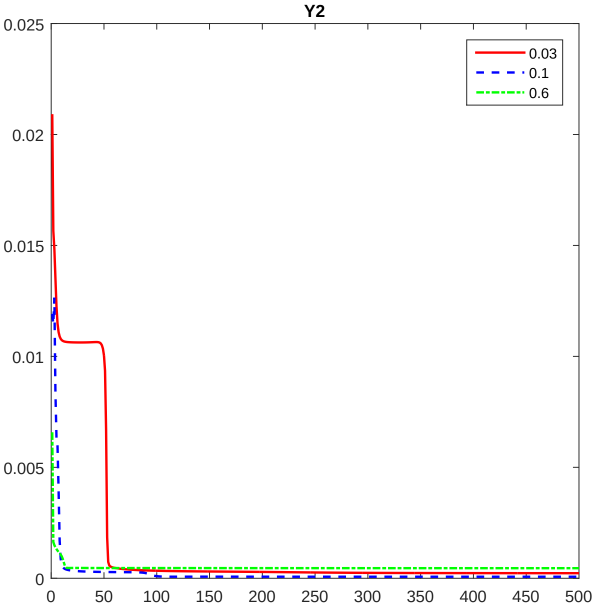




3. For the random weight assignment, use the same initial weights that you obtain the best classification result to compare the learning curves with different learning rates:

   learning rate of 0.03, 0.1, 0.6 are used





Discuss how to obtain a better result according to your experience in this homework:

To obtain a better result, learning rate, epoch, data sequence and the number of nodes in the hidden layers are all important.

1. learning rate: The lower the learning rate is, the better the result will be. But noted that the lower the learning rate is, the more epochs are needed to achieve the convergent result.

2. epoch: For some data, a large number of epochs are needed to converge. Thus, it is better to have more epochs.

3. data sequence: Sometimes, the training data are in an order. It is necessary to randomized the sequence of the training data. By doing this, the convergent time will descend.

4. the number of nodes in the hidden layers: When dealing with simple classification or function approximation problem, nodes between 3 to 5 can provide good results than too many nodes or too little nodes.