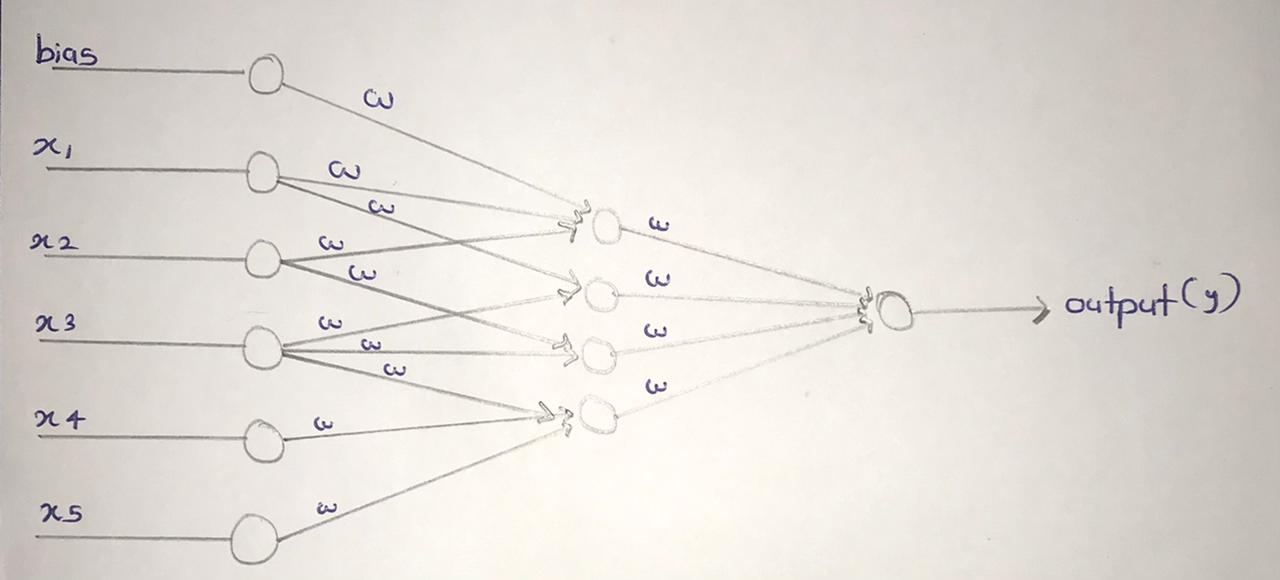
Pre lab :-

* Each weight value's sum product with its corresponding input value and bias value is how all input data in the input layer is calculated.
* Then, the sigmoid function is used to the net inputs to determine the subsequent output. The sigmoid function transmits a value between 0 and 1.
* The overall error is then calculated by adding the errors for each output neuron using the squared error function.
* Next, reduce error throughout the network as a whole and for each output neuron.
* Then, We need to calculate the change in total errors in the outputs.
* The output change will then be calculated once we propagate backward even farther.
* Add up all the data to get the revised weight value and to obtain new weights, we can repeat this process.
* Once we have updated weights guiding the hidden layer neurons, we can inspect the actual updates in the neural network.
* Finally, new values are calculated as we continue the backward pass.
* Question 1



* Question 2

Answers –

dataSet = dataSet(2:end,3:end)

fullDataSet = [1:8143]

trainingData = randperm(8143,5700)

testData = setdiff(fullDataSet,trainingData)

# trainingDataSet

for k = 1:size(trainingData,2)

row = trainingData(k)

if k == 1

trainingDataSet = [dataSet(row,:)];

else

trainingDataSet = [trainingDataSet;dataSet(row,:)];

endif

endfor

# testDataSet

for k = 1:size(testData,2)

row = testData(k)

if k == 1

testDataSet = [dataSet(row,:)];

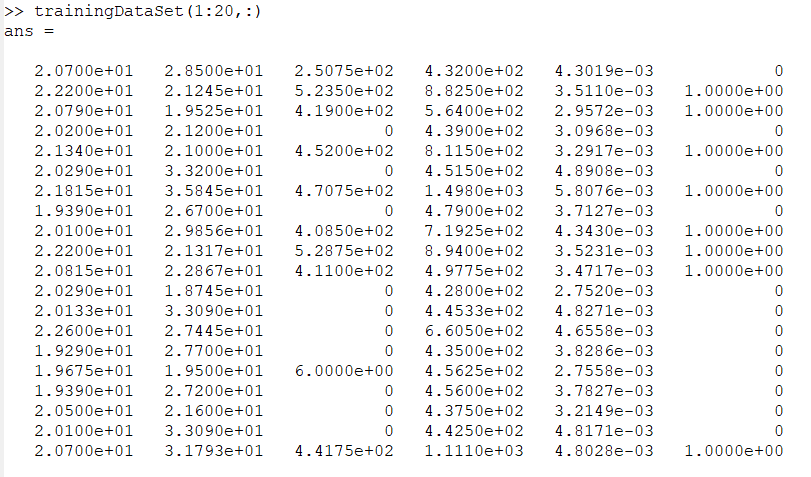
else

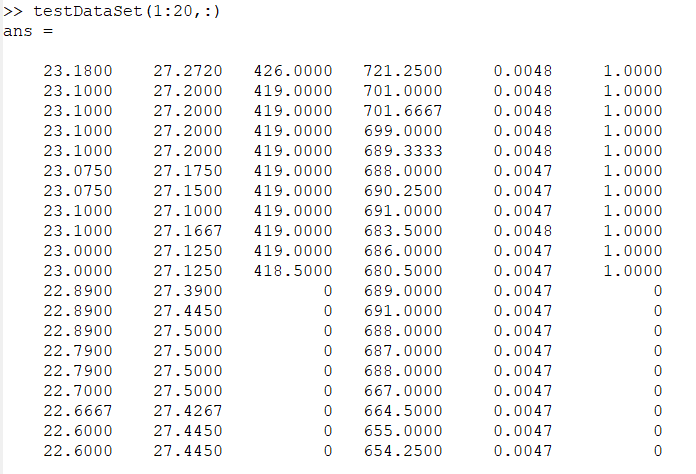
testDataSet = [testDataSet;dataSet(row,:)];

endif

endfor

Results –





* Question 3 part 1

Answers –

function [T1,T2,M2] = OL(trainingData,T1,T2,LR,L,M2=[])

for i=1:L

m=rows(trainingData);

D=0;

for i=1:rows(trainingData)

P1 = [trainingData(i,1:6)'];

R1 = T1\*P1;

#call sigmoid function

P2 = sigmoid(R1);

R2 = T2\*P2;

#call sigmoid function

Z = sigmoid(R2);

B = trainingData(i,7);

error = B-Z;

Q = ((1/2)\*(B-Z)^2);

D=D+Q;

F = error\*Z\*(1-Z);

E = (((T2') \* F ) .\* (P2 .\* (1-P2)));

T1 = T1 - (LR \* (E \*[P1']));

T2 = T2 - (LR \* (F \*[P2']));

endfor

D=D/m;

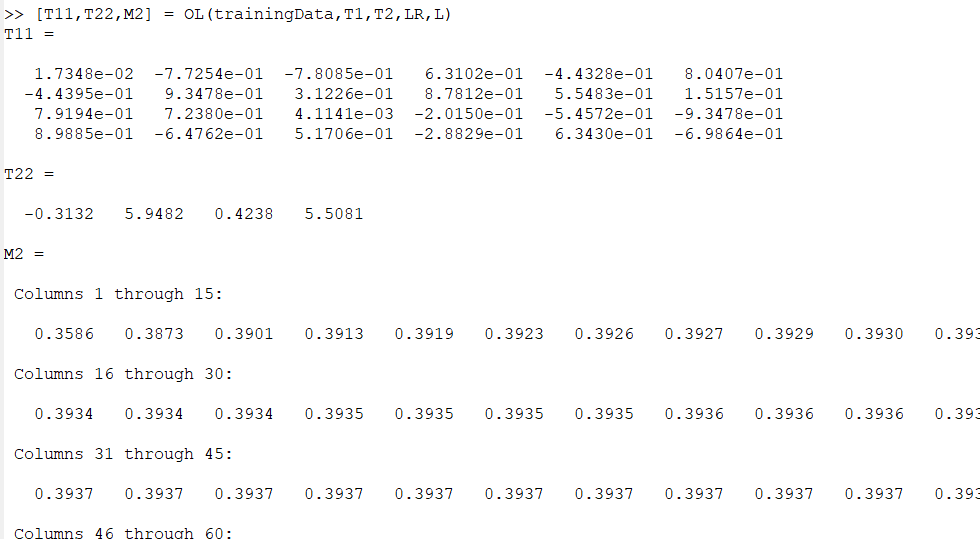
M2=[M2,D];

endfor

endfunction

[T11,T22,M2] = OL(trainingData,T1,T2,LR,L)

Results –



* Question 3 part 2

Answers –

function [T1,T2,M1] = BL(trainingData,T1,T2,LR,L,M1=[])

for i=1:L

m=rows(trainingData);

T2\_Delta=0;

T1\_Delta=0;

D=0;

for i=1:rows(trainingData)

P1 = [trainingData(i,1:6)'];

R1 = T1\*P1;

#call sigmoid function

P2 = sigmoid(R1);

R2 = T2\*P2;

#call sigmoid function

Z = sigmoid(R2);

B = trainingData(i,7);

error = B-Z;

Q = ((1/2)\*(B-Z)^2);

D=D+Q;

F = error\*Z\*(1-Z);

E = (((T2') \* F ) .\* (P2 .\* (1-P2)));

T2\_Delta=T2\_Delta+[F\*P2'];

T1\_Delta=T1\_Delta+[E\*P1'];

endfor

D=D/m;

M1=[M1,D];

T1 = T1 - (LR \* (T1\_Delta/m));

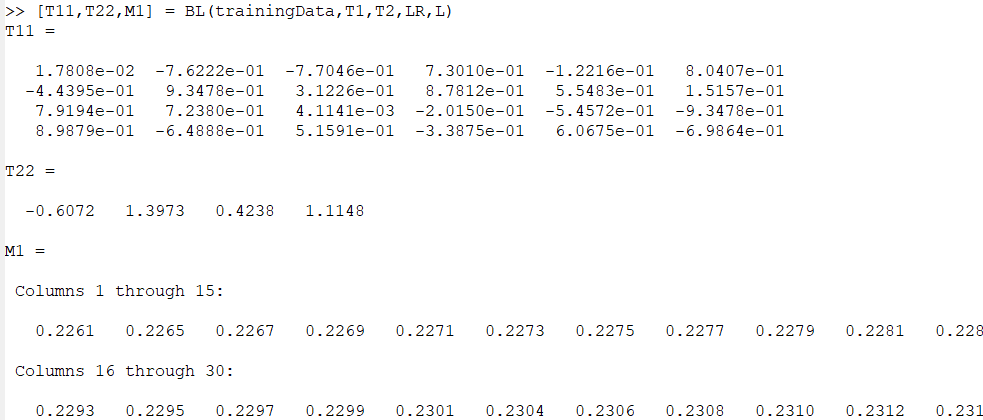
T2 = T2 - (LR \* (T2\_Delta/m));

endfor

endfunction

[T11,T22,M1] = BL(trainingData,T1,T2,LR,L)

Results –



* Question 4

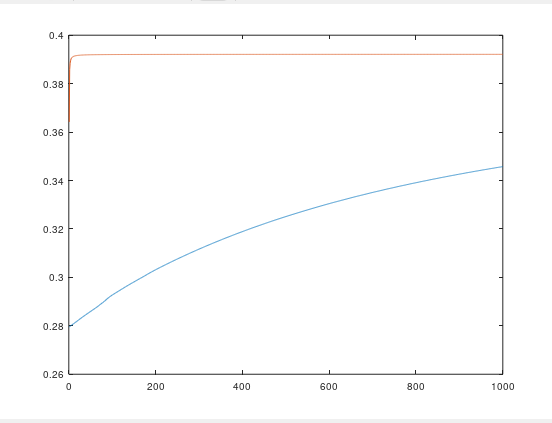
Answers –

plot(k,M1)

hold on

plot(k,M2)

Results –



* Question 5

Answers –

function test\_function(testData,T1,T2)

for i=1:rows(testData)

P1 = [testData(i,1:6)'];

R1 = T1\*P1;

P2 = sigmoid(R1);

R2 = T2\*P2;

Z = sigmoid(R2);

#check threshold value for classification

if Z>0.5

Z=1;

else

Z=0;

endif

Z

B = testData(i,7);

disp("end 1 row")

endfor

endfunction

test\_function(testData,T11,T22)

Results –

