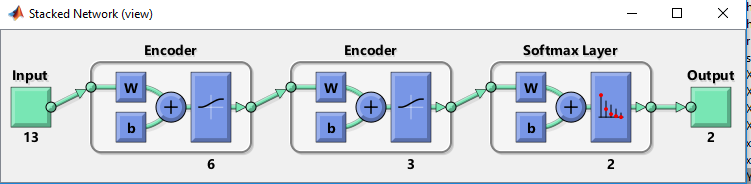
**solutions**

1: Describe your Autoencoder architecture in details (i.e How many layers do you have? How many neurons in each layer? What is the dimensions of the code you used in classification? Did you change anything else in the Autoencoder to increase the prediction quality?

Solution: The architecture used is shown below, with 3 layers, added a Softmax layer at the end. Have tried changing the number of epochs to improve the model.

Layer 1: 13 inputs \* 6 layers = 78 neurons

Layer 2:



2: What is the accuracy, Precision, and Recall for each class prediction?

Solution:

Class 1 (Not interesting bacteria)

Accuracy = 90.9%

Recall = 97.2 %

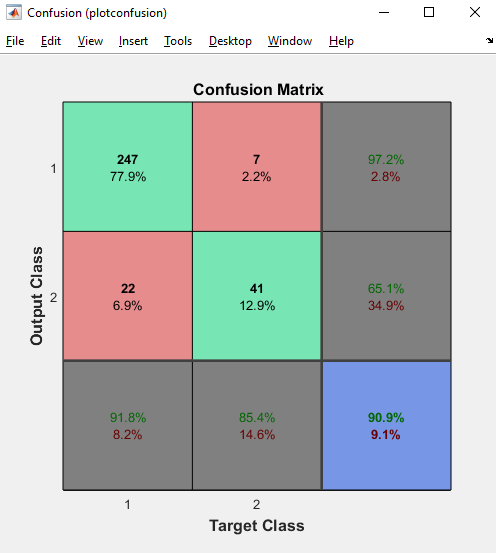
Precision:91.8%

Class 2 (Interesting bacteria)

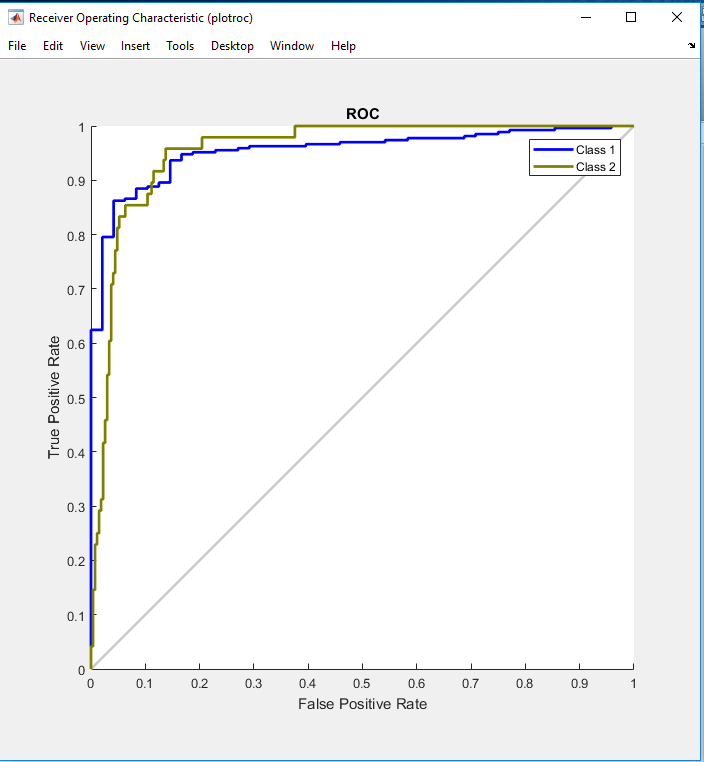
Accuracy = 90.9%

Recall = 65.1 %

Precision:85.4%



3: Create an ROC curve plot for each class prediction



**Matlab Code**

%Load DNA Data

cellDNAData= xlsread('C:\Users\Sai\Desktop\saint thomas classes ppts\Machine Learning\CellDNA.xls');

%0 = bacteria not interested 0, 1 = bacteria interested

for r = 1:length(cellDNAData)

if cellDNAData(r,14) > 0

cellDNAData(r,14) = 1;

else

cellDNAData(r,14) = 0;

end

end

%Convert Data according to Neural network Lib

cellDNAData(:,[15])=cellDNAData(:,14);

for r = 1:length(cellDNAData)

if cellDNAData(r,14) == 0

cellDNAData(r,14) = 1;

else

cellDNAData(r,14) = 0;

end

end

%divide data into train and test

[xTrain,xTest] = dividerand(cellDNAData',0.7,0.3);

xTrain=xTrain';

xTest=xTest';

X = xTrain(:,1:13); Y = xTrain(:,14:15);

%Standardize all predicators In train data

ZX=zscore(X);

X\_Test = xTest(:,1:13); Y\_Test = xTest(:,14:15);

%Standardize all predicators In test data

X\_ZTest=zscore(X\_Test);

X\_ZTest=X\_ZTest';Y\_Test=Y\_Test';

ZX=ZX';

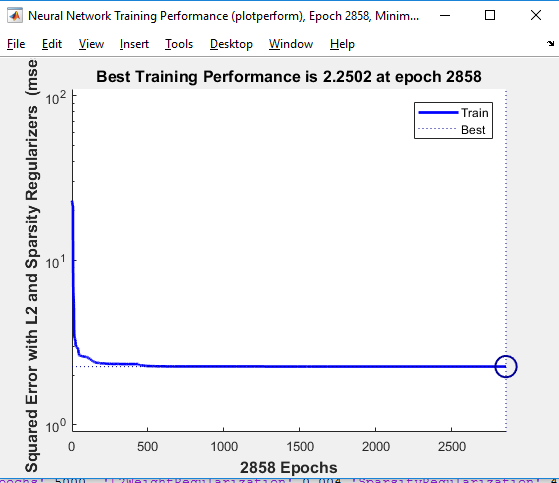
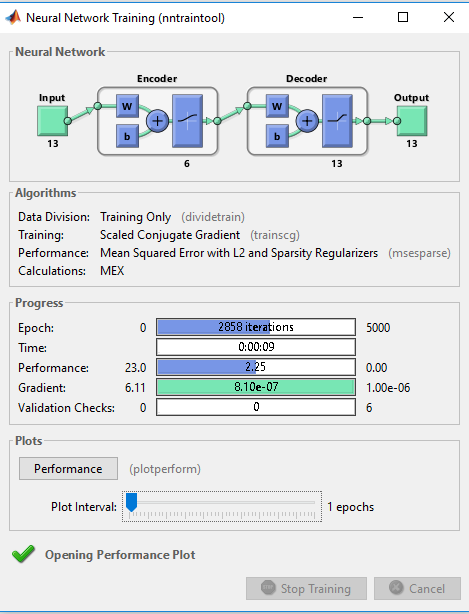
Y=Y';

hiddenSize1=6;

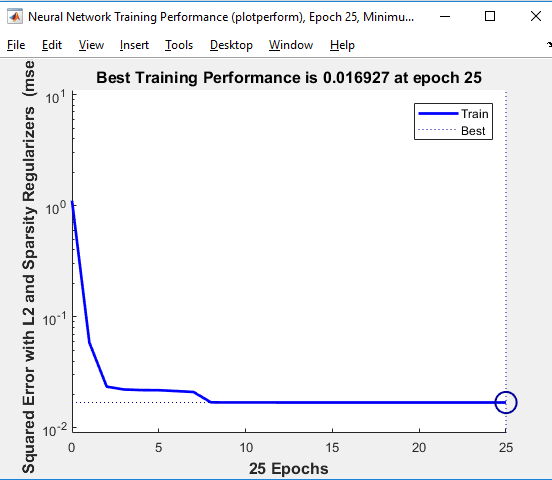
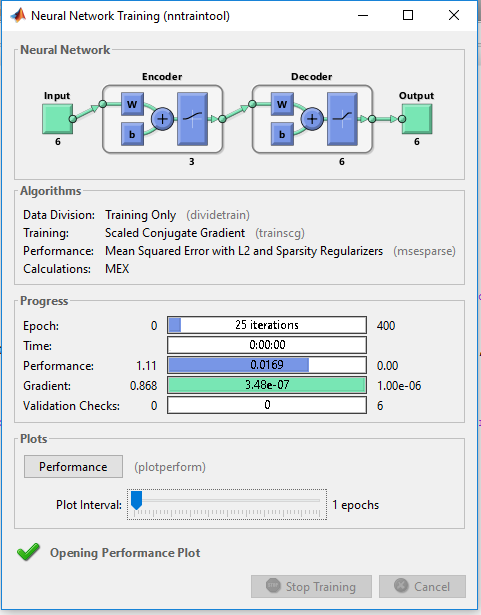
autoenc1 = trainAutoencoder(ZX, hiddenSize1, 'MaxEpochs',5000, 'L2WeightRegularization',0.004,'SparsityRegularization',4, 'SparsityProportion',0.15, 'DecoderTransferFunction','satlin','ScaleData',true);

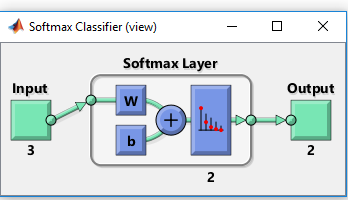
feat1 = encode(autoenc1, ZX);

view(autoenc1)

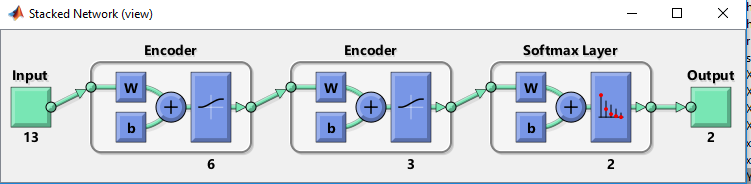


hiddenSize2=3;  
autoenc2 = trainAutoencoder(feat1, hiddenSize2, 'MaxEpochs',400, 'DecoderTransferFunction','satlin','L2WeightRegularization',0.004, 'SparsityRegularization',4, 'SparsityProportion',0.15, 'ScaleData', false);  
feat2 = encode(autoenc2, feat1);  
softnet = trainSoftmaxLayer(feat2, Y, 'MaxEpochs', 400);





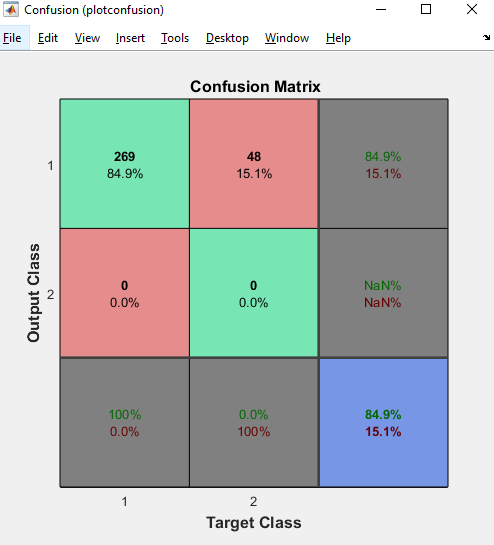
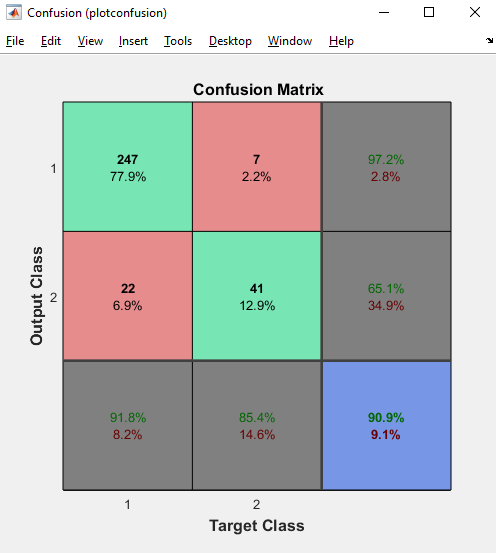
deepnet = stack(autoenc1, autoenc2, softnet);  
view(deepnet);



%Predict the data with autoencoder network & Retrain

yh = deepnet(X\_ZTest);  
plotconfusion(Y\_Test, yh);

%Retrain network  
deepnet = train(deepnet, ZX, Y);  
%Predict after re-training  
yh = deepnet(X\_ZTest);  
plotconfusion(Y\_Test, yh);

Initial Accuracy -----🡪 Retrain --🡪 New Accuracy

Yh\_1 = sim(deepnet,X\_ZTest);

plotroc(Y\_Test,Yh\_1)

