ALLEGATO 4

```
inputSeries = tonndata(input,false,false);
targetSeries = tonndata(target,false,false);
success = 0;

while true
    inputDelays = 1:5;
    feedbackDelays = 1:5;
    hiddenLayerSize = 16;
    net = narxnet(inputDelays, feedbackDelays, hiddenLayerSize);

    net.inputs{1}.processFcns = {'removeconstantrows','mapminmax'};
    net.inputs{2}.processFcns = {'removeconstantrows','mapminmax'};

    [inputs,inputStates,layerStates,targets] = preparets(net,inputSeries,{},↙
targetSeries);

    net.divideFcn = 'dividerand';  % Divide data randomly
    net.divideMode = 'value';      % Divide up every value

    net.divideParam.trainRatio = 70/100;
    net.divideParam.valRatio = 15/100;
    net.divideParam.testRatio = 15/100;

    net.trainFcn = 'trainlm';  % Levenberg-Marquardt
    net.performFcn = 'mse';

    net.plotFcns = {'plotperform','plottrainstate','plotresponse', ...
    'ploterrcorr', 'plotinerrcorr'};

    [net,tr] = train(net,inputs,targets,inputStates,layerStates);

    % Test the Network
    outputs = net(inputs,inputStates,layerStates);
    errors = gsubtract(targets,outputs);
    performance = perform(net,targets,outputs);

    % Recalculate Training, Validation and Test Performance
    trainTargets = gmultiply(targets,tr.trainMask);
    valTargets = gmultiply(targets,tr.valMask);
    testTargets = gmultiply(targets,tr.testMask);
    trainPerformance = perform(net,trainTargets,outputs);
    valPerformance = perform(net,valTargets,outputs);
    testPerformance = perform(net,testTargets,outputs);

    % Closed Loop Network
    % Use this network to do multi-step prediction.
    % The function CLOSELOOP replaces the feedback input with a direct
    % connection from the outout layer.
    netc = closeloop(net);
    netc.name = [net.name ' - Closed Loop'];
    [xc,xic,aic,tc] = preparets(netc,inputSeries,{},targetSeries);
    yc = netc(xc,xic,aic);
    closedLoopPerformance = perform(netc,tc,yc);

    % Early Prediction Network
    % For some applications it helps to get the prediction a timestep early.
    % The original network returns predicted y(t+1) at the same time it is given y↙
(t+1).
    % For some applications such as decision making, it would help to have predicted
    % y(t+1) once y(t) is available, but before the actual y(t+1) occurs.
    % The network can be made to return its output a timestep early by removing one↙
delay
    % so that its minimal tap delay is now 0 instead of 1.  The new network returns↙
the
    % same outputs as the original network, but outputs are shifted left one timestep.
    nets = removedelay(net);
    nets.name = [net.name ' - Predict One Step Ahead'];
    [xs,xis,ais,ts] = preparets(nets,inputSeries,{},targetSeries);
    ys = nets(xs,xis,ais);
    earlyPredictPerformance = perform(nets,ts,ys);

    if (valPerformance < 100)
        success = success + 1;
        mkdir(strcat('/Users/stefy/Dropbox/matlaboutput/',int2str(success)));
        saveas(plotperform(tr),strcat('/Users/stefy/Dropbox/matlaboutput/',int2str↙
(success),'/performance.png'),'png');
        saveas(plotregression(targets,outputs),strcat↙
('/Users/stefy/Dropbox/matlaboutput/',int2str(success),'/regression.png'),'png');
        saveas(plotresponse(targets,outputs),strcat↙
('/Users/stefy/Dropbox/matlaboutput/',int2str(success),'/response.png'),'png');
        saveas(ploterrcorr(errors),strcat('/Users/stefy/Dropbox/matlaboutput/',int2str↙
(success),'/errorcorrelation.png'),'png');
        saveas(ploterrhist(errors),strcat('/Users/stefy/Dropbox/matlaboutput/',int2str↙
(success),'/errohistogram.png'),'png');
        saveas(plotinerrcorr(inputs,errors),strcat↙
('/Users/stefy/Dropbox/matlaboutput/',int2str(success),'/inputerrorcorrelation.↙
png'),'png');
        save(strcat('/Users/stefy/Dropbox/matlaboutput/',int2str(success),'/workspace.↙
mat'));
        %break;
    end;
    %risultati(i,j) = valPerformance;
    %textout = ['Best validation performance: ', num2str(risultati(i,j))];
    %disp(textout);
end

%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, plotregression(targets,outputs)
%figure, plotresponse(targets,outputs)
%figure, ploterrcorr(errors)
%figure, ploterrhist(errors)
%figure, plotinerrcorr(inputs,errors)

disp('Script terminato');
```