**Name: Yezhou(Lucy) CHEN**

**ID: 222016601105035**

## 1. Introduction:

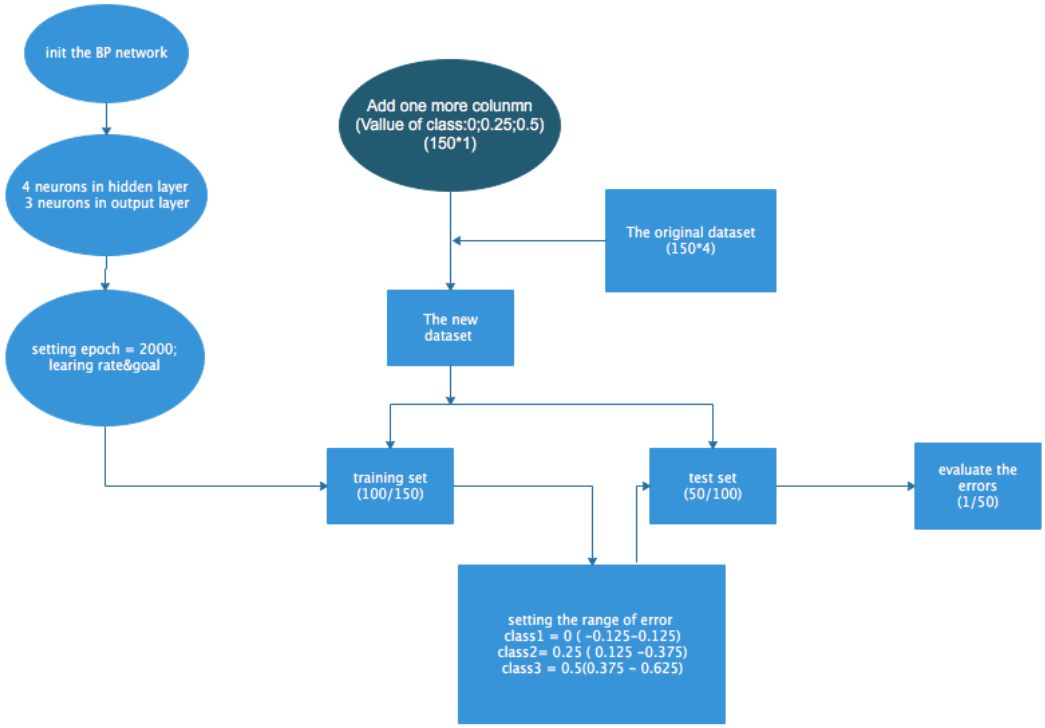
The iris plant data set has 150 elements total, 3 classes of 50 elements each, where each class refers to a type of the Iris plant. Each plant is characterized by 4 factors. That is sepal length, sepal width, petal length and petal width.

## 2. The short description of the domain problem:

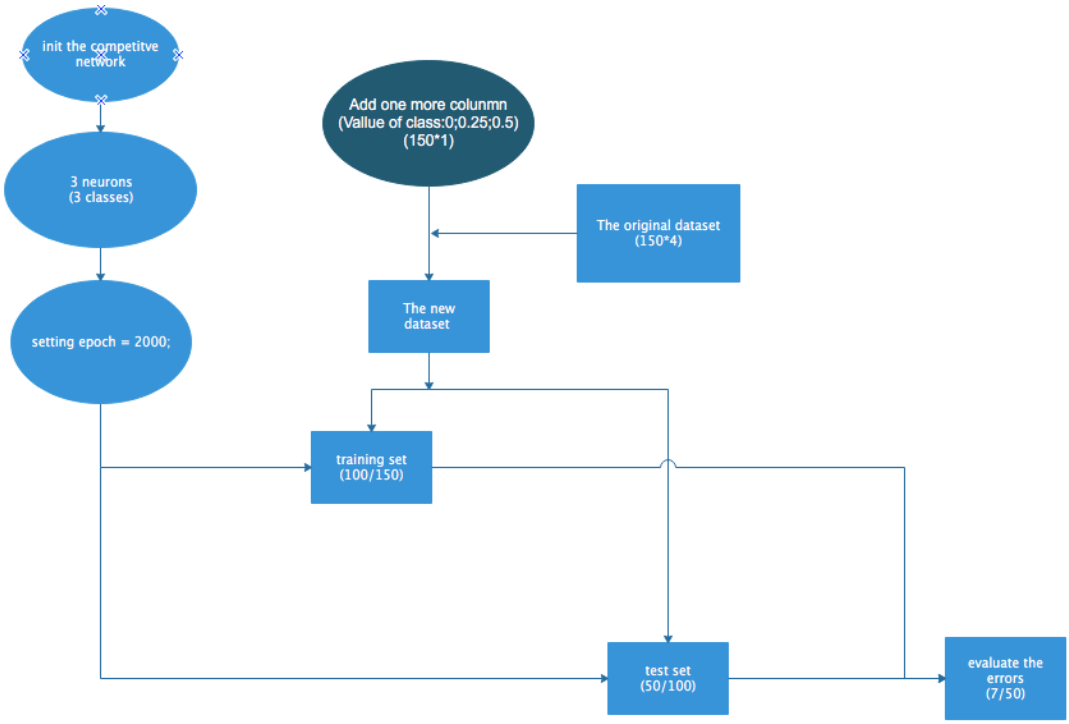
the domain problem is to classify the iris plants via artificial neural networks. That is to say, utilizing the 4 factors, or the 4 values in the set to make sure which class this iris plant is in, by a three-layer-back-propagation neural network and a single-layer competitive network is my work in this assignment.

## 3. The overview of the developed program (flow chart)

(1) Three-layer-back-propagation neural network




(2) single-layer competitive network




**4. Comparison of the performance of a three-layer back-propagation neural network with a single-layer competitive network and discussion of the results.**

(1) Implementation and Testing (part 1)

```
iris_data = [  5.1 3.5 1.4 0.2  % Iris-setosa
               4.9 3.0 1.4 0.2   % Iris-setosa
               4.7 3.2 1.3 0.2   % Iris-setosa
               4.6 3.1 1.5 0.2   % Iris-setosa
               5.0 3.6 1.4 0.2   % Iris-setosa
               5.4 3.9 1.7 0.4   % Iris-setosa
               4.6 3.4 1.4 0.3   % Iris-setosa
               5.0 3.4 1.5 0.2   % Iris-setosa
               4.4 2.9 1.4 0.2   % Iris-setosa
               4.9 3.1 1.5 0.1   % Iris-setosa
               5.4 3.7 1.5 0.2   % Iris-setosa
               4.8 3.4 1.6 0.2   % Iris-setosa
               4.8 3.0 1.4 0.1   % Iris-setosa
               4.3 3.0 1.1 0.1   % Iris-setosa
               5.8 4.0 1.2 0.2   % Iris-setosa
               5.7 4.4 1.5 0.4   % Iris-setosa
               5.4 3.9 1.3 0.4   % Iris-setosa
               5.1 3.5 1.4 0.3   % Iris-setosa
               5.7 3.8 1.7 0.3   % Iris-setosa
               5.1 3.8 1.5 0.3   % Iris-setosa
               5.4 3.4 1.7 0.2   % Iris-setosa
               5.1 3.7 1.5 0.4   % Iris-setosa
               4.6 3.6 1.0 0.2   % Iris-setosa
               5.1 3.3 1.7 0.5   % Iris-setosa
               4.8 3.4 1.9 0.2   % Iris-setosa
               5.0 3.0 1.6 0.2   % Iris-setosa
               5.0 3.4 1.6 0.4   % Iris-setosa
               5.2 3.5 1.5 0.2   % Iris-setosa
               5.2 3.4 1.4 0.2   % Iris-setosa
               4.7 3.2 1.6 0.2   % Iris-setosa
               4.8 3.1 1.6 0.2   % Iris-setosa
               5.4 3.4 1.5 0.4   % Iris-setosa
```

```
5.2 4.1 1.5 0.1   % Iris-setosa
5.5 4.2 1.4 0.2   % Iris-setosa
4.9 3.1 1.5 0.1   % Iris-setosa
5.0 3.2 1.2 0.2   % Iris-setosa
5.5 3.5 1.3 0.2   % Iris-setosa
4.9 3.1 1.5 0.1   % Iris-setosa
4.4 3.0 1.3 0.2   % Iris-setosa
5.1 3.4 1.5 0.2   % Iris-setosa
5.0 3.5 1.3 0.3   % Iris-setosa
4.5 2.3 1.3 0.3   % Iris-setosa
4.4 3.2 1.3 0.2   % Iris-setosa
5.0 3.5 1.6 0.6   % Iris-setosa
5.1 3.8 1.9 0.4   % Iris-setosa
4.8 3.0 1.4 0.3   % Iris-setosa
5.1 3.8 1.6 0.2   % Iris-setosa
4.6 3.2 1.4 0.2   % Iris-setosa
5.3 3.7 1.5 0.2   % Iris-setosa
5.0 3.3 1.4 0.2   % Iris-setosa
7.0 3.2 4.7 1.4   % Iris-versicolor
6.4 3.2 4.5 1.5   % Iris-versicolor
6.9 3.1 4.9 1.5   % Iris-versicolor
5.5 2.3 4.0 1.3   % Iris-versicolor
6.5 2.8 4.6 1.5   % Iris-versicolor
5.7 2.8 4.5 1.3   % Iris-versicolor
6.3 3.3 4.7 1.6   % Iris-versicolor
4.9 2.4 3.3 1.0   % Iris-versicolor
6.6 2.9 4.6 1.3   % Iris-versicolor
5.2 2.7 3.9 1.4   % Iris-versicolor
5.0 2.0 3.5 1.0   % Iris-versicolor
5.9 3.0 4.2 1.5   % Iris-versicolor
6.0 2.2 4.0 1.0   % Iris-versicolor
6.1 2.9 4.7 1.4   % Iris-versicolor
5.6 2.9 3.6 1.3   % Iris-versicolor
6.7 3.1 4.4 1.4   % Iris-versicolor
5.6 3.0 4.5 1.5   % Iris-versicolor
5.8 2.7 4.1 1.0   % Iris-versicolor
6.2 2.2 4.5 1.5   % Iris-versicolor
5.6 2.5 3.9 1.1   % Iris-versicolor
5.9 3.2 4.8 1.8   % Iris-versicolor
6.1 2.8 4.0 1.3   % Iris-versicolor
6.3 2.5 4.9 1.5   % Iris-versicolor
6.1 2.8 4.7 1.2   % Iris-versicolor
6.4 2.9 4.3 1.3   % Iris-versicolor
6.6 3.0 4.4 1.4   % Iris-versicolor
6.8 2.8 4.8 1.4   % Iris-versicolor
6.7 3.0 5.0 1.7   % Iris-versicolor
6.0 2.9 4.5 1.5   % Iris-versicolor
5.7 2.6 3.5 1.0   % Iris-versicolor
5.5 2.4 3.8 1.1   % Iris-versicolor
5.5 2.4 3.7 1.0   % Iris-versicolor
5.8 2.7 3.9 1.2   % Iris-versicolor
6.0 2.7 5.1 1.6   % Iris-versicolor
5.4 3.0 4.5 1.5   % Iris-versicolor
6.0 3.4 4.5 1.6   % Iris-versicolor
6.7 3.1 4.7 1.5   % Iris-versicolor
6.3 2.3 4.4 1.3   % Iris-versicolor
5.6 3.0 4.1 1.3   % Iris-versicolor
5.5 2.5 4.0 1.3   % Iris-versicolor
5.5 2.6 4.4 1.2   % Iris-versicolor
6.1 3.0 4.6 1.4   % Iris-versicolor
5.8 2.6 4.0 1.2   % Iris-versicolor
5.0 2.3 3.3 1.0   % Iris-versicolor
5.6 2.7 4.2 1.3   % Iris-versicolor
5.7 3.0 4.2 1.2   % Iris-versicolor
5.7 2.9 4.2 1.3   % Iris-versicolor
6.2 2.9 4.3 1.3   % Iris-versicolor
5.1 2.5 3.0 1.1   % Iris-versicolor
5.7 2.8 4.1 1.3   % Iris-versicolor
6.3 3.3 6.0 2.5   % Iris-verginica
5.8 2.7 5.1 1.9   % Iris-verginica
7.1 3.0 5.9 2.1   % Iris-verginica
6.3 2.9 5.6 1.8   % Iris-verginica
6.5 3.0 5.8 2.2   % Iris-verginica
7.6 3.0 6.6 2.1   % Iris-verginica
4.9 2.5 4.5 1.7   % Iris-verginica
7.3 2.9 6.3 1.8   % Iris-verginica
6.7 2.5 5.8 1.8   % Iris-verginica
```

```
    7.2 3.6 6.1 2.5   % Iris-verginica
    6.5 3.2 5.1 2.0   % Iris-verginica
    6.4 2.7 5.3 1.9   % Iris-verginica
    6.8 3.0 5.5 2.1   % Iris-verginica
    5.7 2.5 5.0 2.0   % Iris-verginica
    5.8 2.8 5.1 2.4   % Iris-verginica
    6.4 3.2 5.3 2.3   % Iris-verginica
    6.5 3.0 5.5 1.8   % Iris-verginica
    7.7 3.8 6.7 2.2   % Iris-verginica
    7.7 2.6 6.9 2.3   % Iris-verginica
    6.0 2.2 5.0 1.5   % Iris-verginica
    6.9 3.2 5.7 2.3   % Iris-verginica
    5.6 2.8 4.9 2.0   % Iris-verginica
    7.7 2.8 6.7 2.0   % Iris-verginica
    6.3 2.7 4.9 1.8   % Iris-verginica
    6.7 3.3 5.7 2.1   % Iris-verginica
    7.2 3.2 6.0 1.8   % Iris-verginica
    6.2 2.8 4.8 1.8   % Iris-verginica
    6.1 3.0 4.9 1.8   % Iris-verginica
    6.4 2.8 5.6 2.1   % Iris-verginica
    7.2 3.0 5.8 1.6   % Iris-verginica
    7.4 2.8 6.1 1.9   % Iris-verginica
    7.9 3.8 6.4 2.0   % Iris-verginica
    6.4 2.8 5.6 2.2   % Iris-verginica
    6.3 2.8 5.1 1.5   % Iris-verginica
    6.1 2.6 5.6 1.4   % Iris-verginica
    7.7 3.0 6.1 2.3   % Iris-verginica
    6.3 3.4 5.6 2.4   % Iris-verginica
    6.4 3.1 5.5 1.8   % Iris-verginica
    6.0 3.0 4.8 1.8   % Iris-verginica
    6.9 3.1 5.4 2.1   % Iris-verginica
    6.7 3.1 5.6 2.4   % Iris-verginica
    6.9 3.1 5.1 2.3   % Iris-verginica
    5.8 2.7 5.1 1.9   % Iris-verginica
    6.8 3.2 5.9 2.3   % Iris-verginica
    6.7 3.3 5.7 2.5   % Iris-verginica
    6.7 3.0 5.2 2.3   % Iris-verginica
    6.3 2.5 5.0 1.9   % Iris-verginica
    6.5 3.0 5.2 2.0   % Iris-verginica
    6.2 3.4 5.4 2.3   % Iris-verginica
    5.9 3.0 5.1 1.8]; % Iris-verginica


%add one more column which could make sure the class
iris_data_2=zeros(150,1);
%case 1
for i = 1:50
   iris_data_2(i)=0;
end

%case 2
for i =51:100
    iris_data_2(i)=0.25;
end


%case 3
for i =101:150
    iris_data_2(i)=0.5;
end

iris_data = [iris_data iris_data_2];


 %training and validation
k = rand(1,150);
[m,n]=sort(k);
input_train= iris_data(n(1:100),(1:4));
output_train = iris_data(n(1:100),5);
input_test= iris_data(n(101:150),(1:4));
output_test= iris_data(n(101:150),5);
```

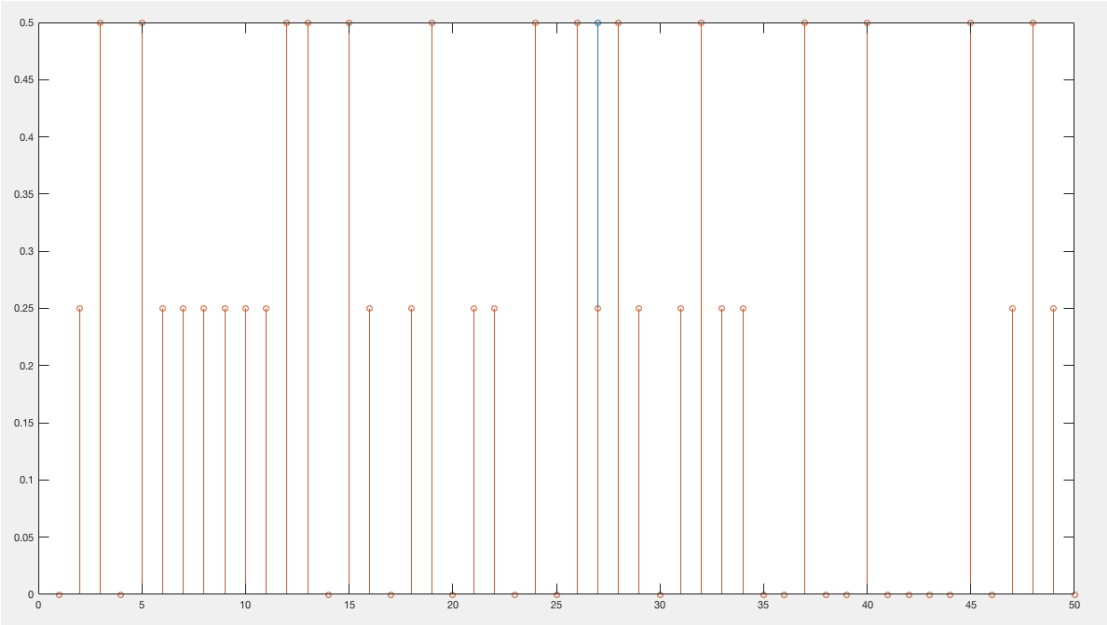
```
          %build the network
          s1 = 4;
          s2 = 3;
          net =
newff(input_train',output_train',s1,{'tansig','purelin'},'traingd');
          net.divideFcn = '';%let all the samples in the training set to be
trained by the neural net work


          net.trainParam.show=1;      % Number of epochs between showing the
progress
          net.trainParam.epochs=2000; % Maximum number of epochs
          net.trainParam.goal=0.004;  % Performance goal
          net.trainParam.lr=0.1;      % Learning rate


          %training & validation process
          [net,tr]=train(net,input_train',output_train');

          an= sim(net, input_test');


          %the range of erroe
          for i=1:50
              if an(i)>=-0.125 && an(i)<0.125
                  an(i)=0;
              end


              if an(i)>=0.125 && an(i)<0.375
                 an(i)= 0.25;
              end

              if an(i)>=0.375 && an(i)<0.625
                  an(i)=0.5;
              end

          end
```

**Test result: (via graph)**

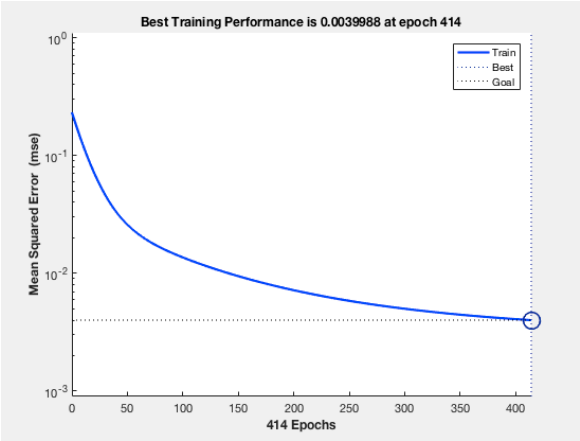
>> stem(an)

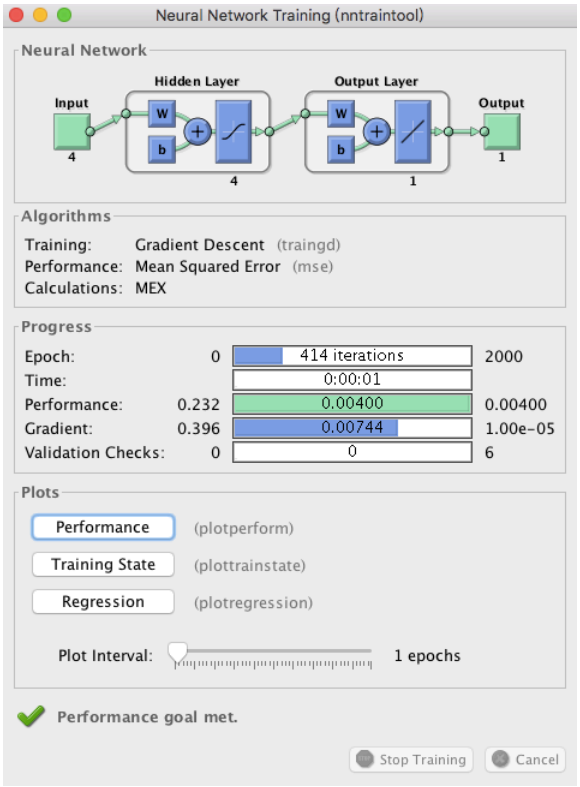
>> hold on

>> stem (output_test')



Through this graph, the recognition error is 1 out of 50. The error rate is 0.02.




(2) Implementation and Testing (part 2)

```
iris_data = [the same as the data set before, in order to make this
assignment report as simple and concise as I can ]
```

```
%case 1
iris_data_2=zeros(150,1);

for i = 1:50
   iris_data_2(i)=1;
end

%case2
for i =51:100
    iris_data_2(i)=2;
end


%case 3
for i =101:150
    iris_data_2(i)=3;
end


iris_data = [iris_data iris_data_2];



%train & validation set

k = rand(1,150);
[m,n]=sort(k);
input_train= iris_data(n(1:100),(1:4));
```

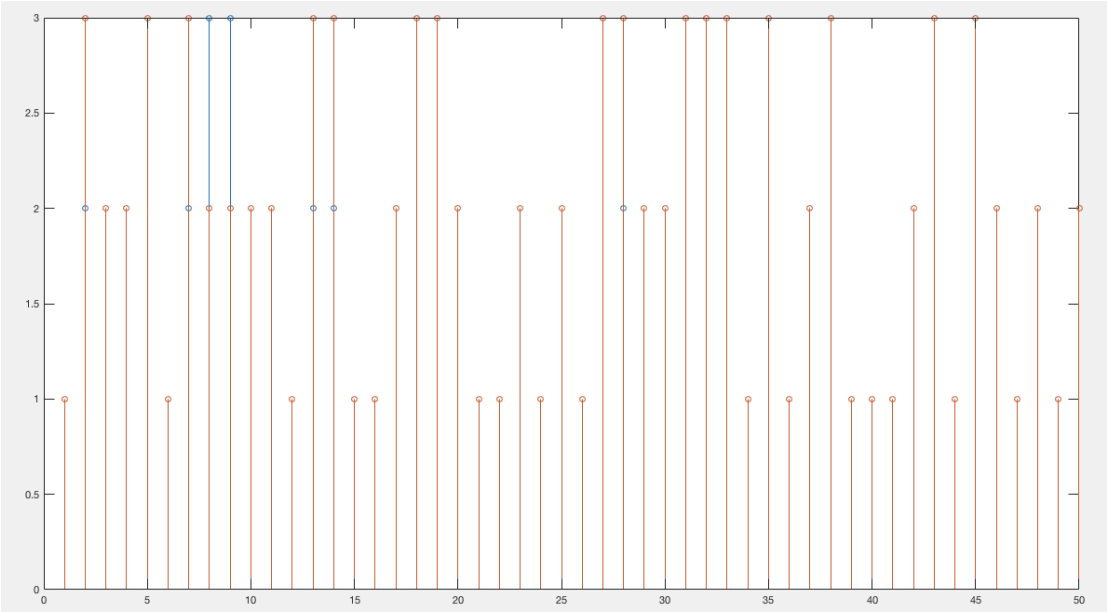
```
output_train = iris_data(n(1:100),5);
input_test= iris_data(n(101:150),(1:4));
output_test= iris_data(n(101:150),5);


%competitve neural network

 s2 = 3;
net = newc(input_train',s2);

net = init(net);


net.trainParam.epochs=2000; % Maximum number of epochs
net.trainParam.lr=0.1;      % Learning rate



%training proces
net=train(net,input_train');
bn = sim(net, input_train');
bnc= vec2ind(bn);

%test(validation) process
an= sim(net, input_test');
anc =vec2ind(an) ;
```

**Test result: (via graph)**

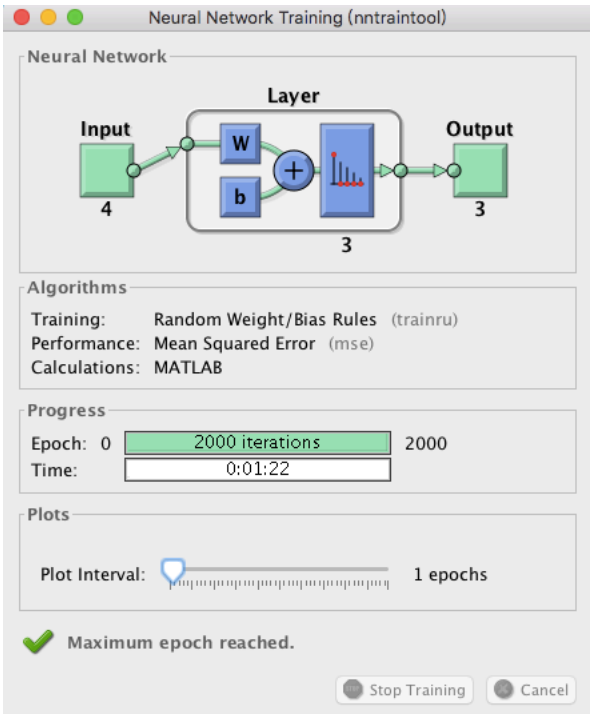
>> stem(anc)

>> hold on

>> stem (output_test')




Through this graph, the recognition error is 7 out of 50. The error rate is 0.14.




Comparison & Conclusion:

1. Why BP network works so well ?

   From my point of view, the answer is the number of neurons in the hidden layer. I set 4 neurons in the hidden layer, because each iris is characterized by 4 factors. Since I read some academic paper which is related to the number of neurons in the hidden layer. The preferable way is to let the number of neurons in hidden layer be same as the number of factors, which is four.

   As you can see, the result is good, only one error among 50 tests.

2. Why competitive neural network not perform as well as the BP network?

   The answer lies in various aspects.

   (1) The competitive neural network in MATLAB Neural Network Toolbox has some disadvantages. I can only set the Maximum number of epochs, Learning rate. But the Performance goal which I can decide in the BP network, I cannot set it in the competitive network. That is to say, I cannot evaluate the number of epochs in the competitive network, no specific rules. The only way for me to decide the number of epochs is intuition.( just like the picture before)

   (2) The performance of this competitive neural networks based on the

learning rate, the number of epochs, the training time and some other factors. But the reality is that there is no efficient method to evaluate these parameters.

(3) For the training set, because I set the random number to select 100 samples from the original data set. Since it is random, the samples which I select to train are always different. Perhaps, the samples are chaotic, no obvious classification features. What is worse, these features maybe quite similar, the competitive cannot perform well due to these two kinds of circumstances.

(4) The way which I estimate the error is to compare stem(anc) and stem(output_test'). But the problem is that some samples, the competitive network could aggregate these sample in one class, but the not the class I set in the output_test. For example, I set `Iris-setosa` the class one. When the competitive network integrate perhaps 7 samples in the class one, the network integrate 5 out of 7 samples to be class 3, the rest 2 becomes class1. So I am wondering does the competitive neural network takes class 3 as class 1, so it only have 2 errors. Or the network takes class 1 as class 1, so it has 5 errors. [I do not know whether I explain my doubt clear or not? I really want to get the answer from you through Wechat].