Using Matlab Neural Networks Toolbox

IMS Class

Most of the sides are from the Matlab tutorial.

Introduction

- Matlab Neural Network Toolbox provides tools for designing, implementing, visualizing, and simulating neural networks.
- It supports feedforward networks, radial basis networks, dynamic networks, self-organizing maps, and other proven network paradigms.
- We will follow Matlab's examples to learn to use four graphical tools for training neural networks to solve problems in function fitting, pattern recognition (clustering, and time series on your own).

Starting the Master GUI

- To start the master GUI type
  - nnstart
- This enables us to access the GUIs for the following tasks
  - Function fitting
  - Pattern recognition
  - Data clustering
  - Time series analysis
- The second way of using the toolbox is through command line operation, which we will not cover.

Neural Network Design Steps

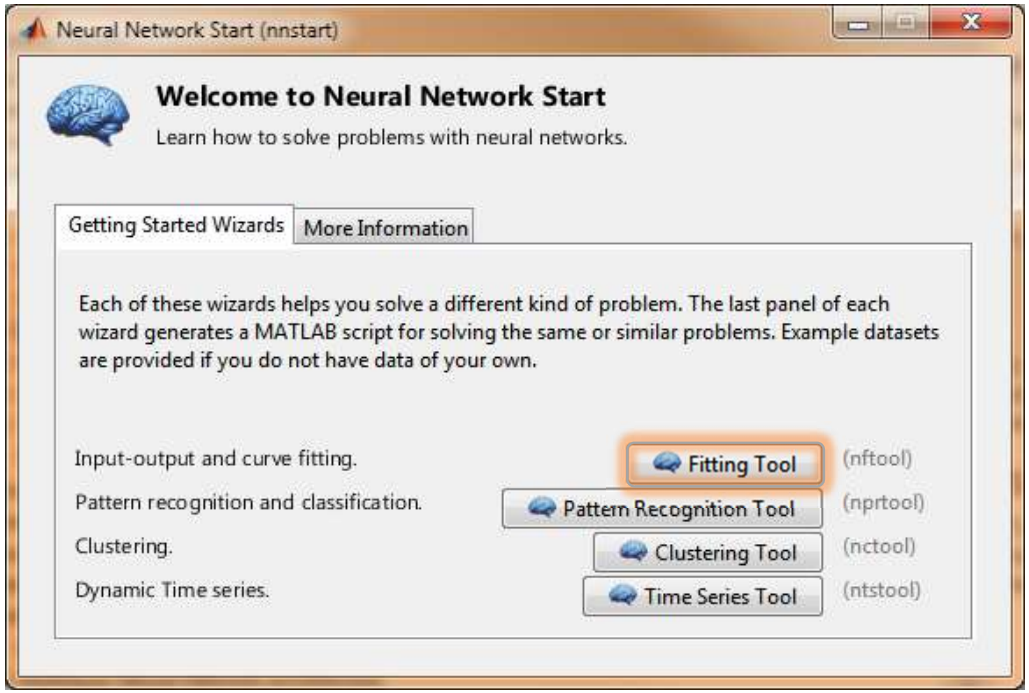
- Standard steps in designing a NN in Matlab are
  1. Collect data
  2. Create the network
  3. Configure the network
  4. Initialize the weights and biases
  5. Train the network
  6. Validate the network
  7. Use the network

Fitting a Function

Problem Definition

- Suppose, for instance, that you have data from a housing application. You want to design a network that can predict the value of a house (in $1000s), given 13 pieces of geographical and real estate information. You have a total of 506 example homes for which you have those 13 items of data and their associated market values

nnstart



- Click Fitting Tool to open the Neural Network Fitting Tool.

Neural Network Fitting Tool (nftool)

Welcome to the Neural Network Fitting Tool.

Solve an input-output fitting problem with a two-layer feed-forward neural network.

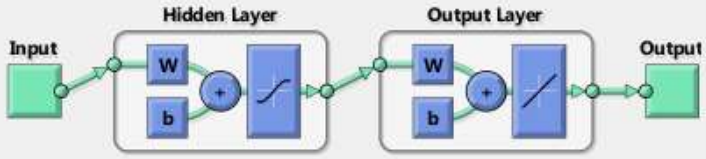
Introduction

In fitting problems, you want a neural network to map between a data set of numeric inputs and a set of numeric targets.

Examples of this type of problem include estimating house prices from such input variables as tax rate, pupil/teacher ratio in local schools and crime rate (house_dataset); estimating engine emission levels based on measurements of fuel consumption and speed (engine_dataset); or predicting a patient's bodyfat level based on body measurements (bodyfat_dataset).

The Neural Network Fitting Tool will help you select data, create and train a network, and evaluate its performance using mean square error and regression analysis.

Neural Network

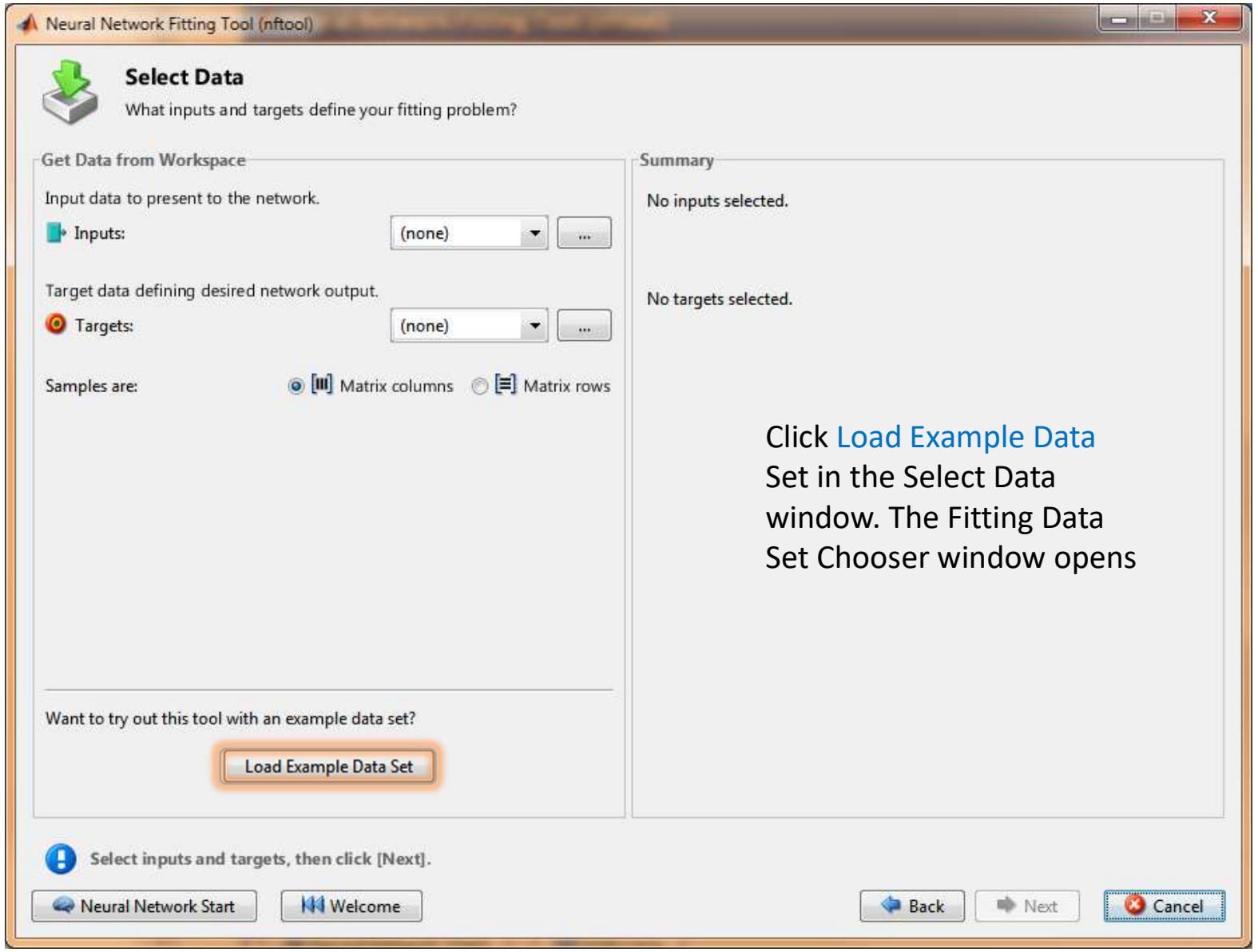


A two-layer feed-forward network with sigmoid hidden neurons and linear output neurons (fitnet), can fit multi-dimensional mapping problems arbitrarily well, given consistent data and enough neurons in its hidden layer.

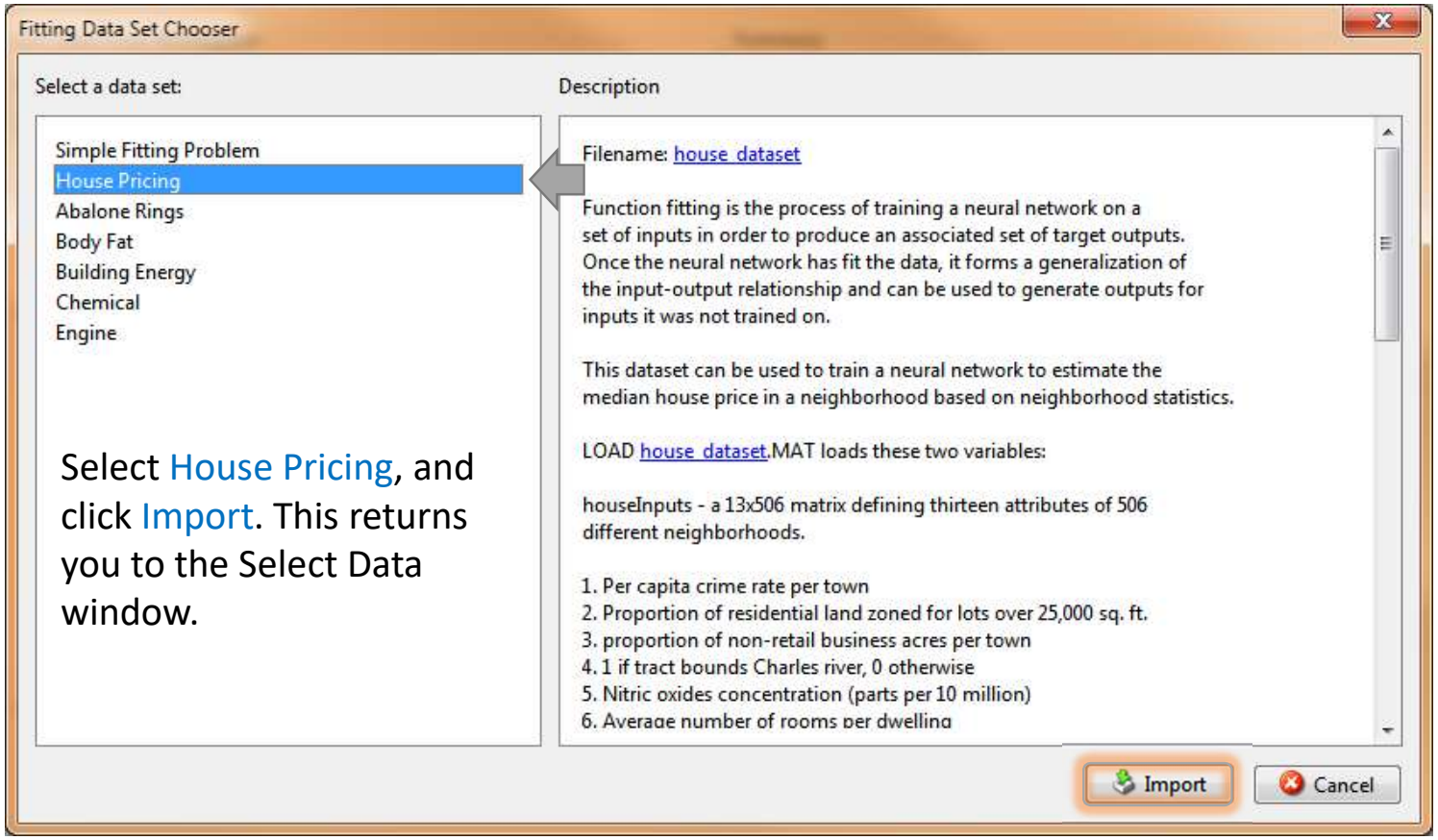
The network will be trained with Levenberg-Marquardt backpropagation algorithm (trainlm), unless there is not enough memory, in which case scaled conjugate gradient backpropagation (trainscg) will be used.

To continue, click [Next].

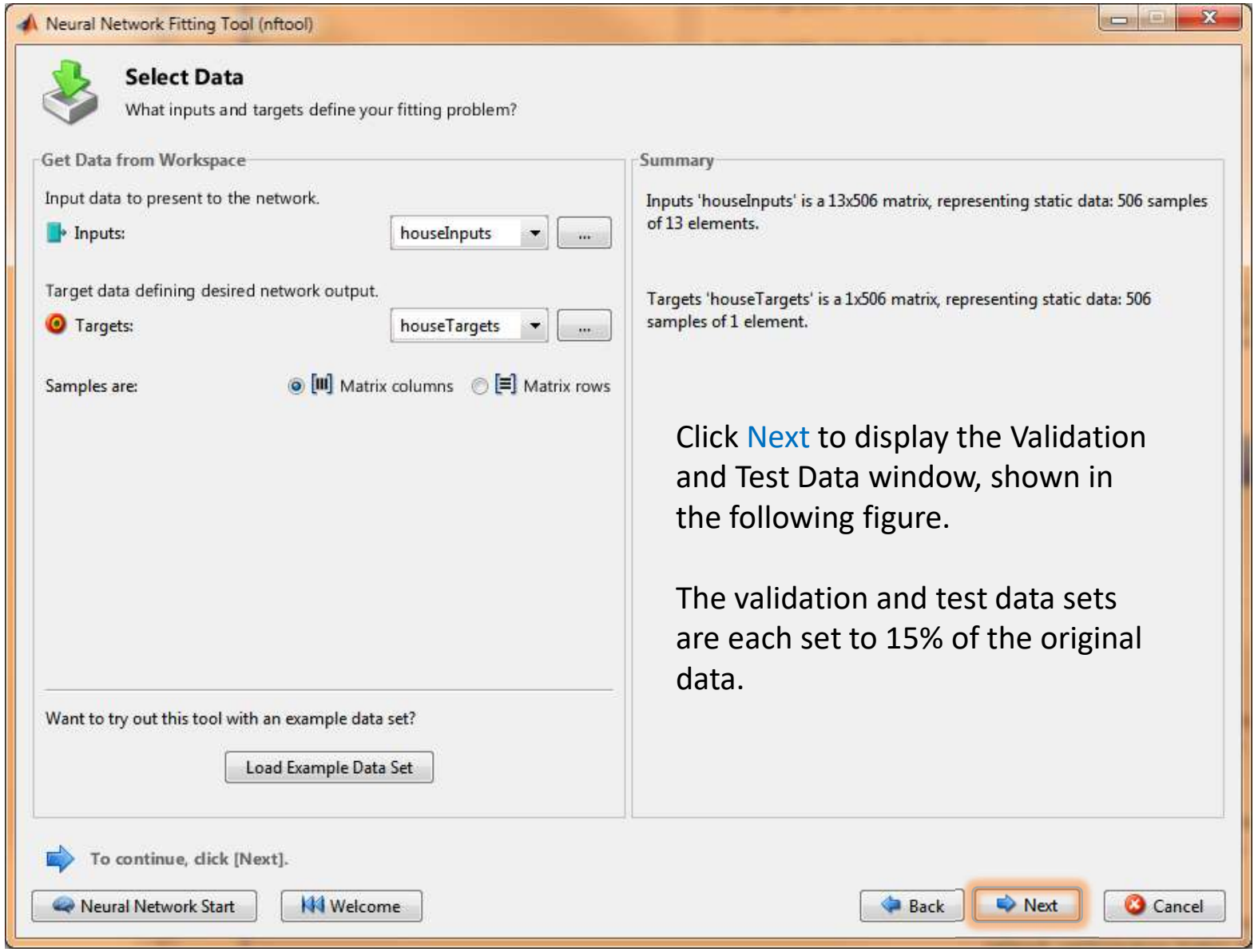
Neural Network Start | Welcome | Back | Next | Cancel



Click Load Example Data Set in the Select Data window. The Fitting Data Set Chooser window opens

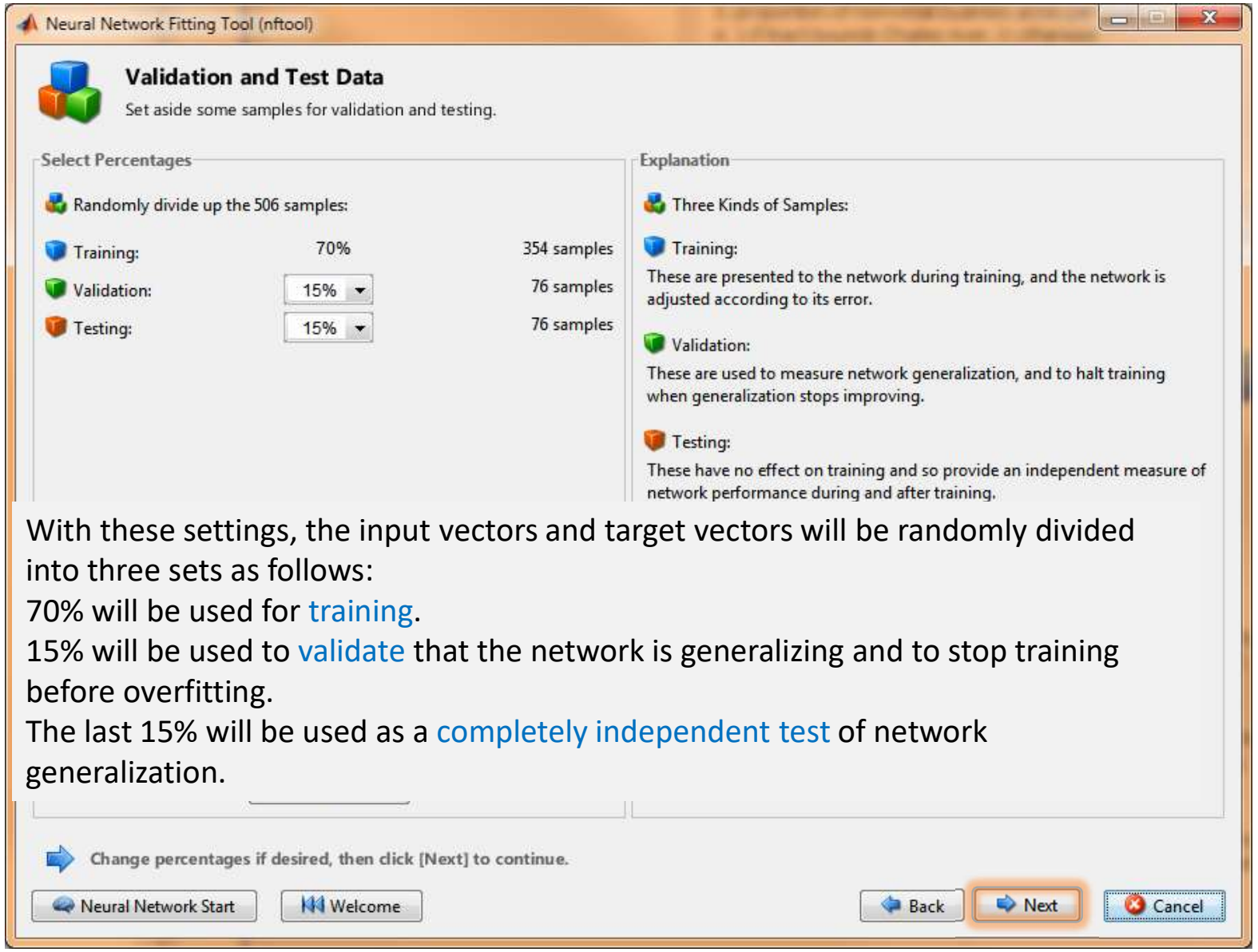


Select House Pricing, and click Import. This returns you to the Select Data window.

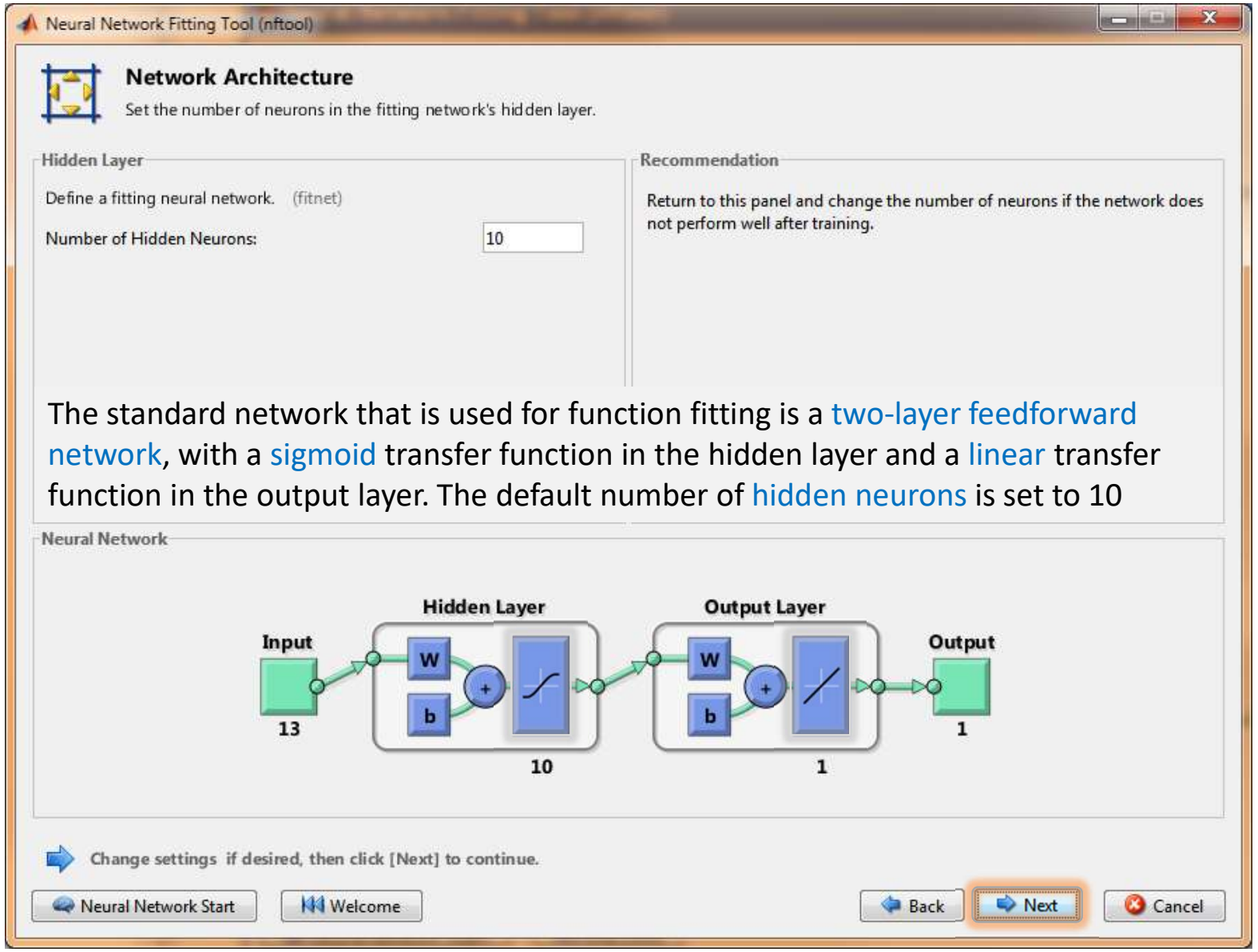


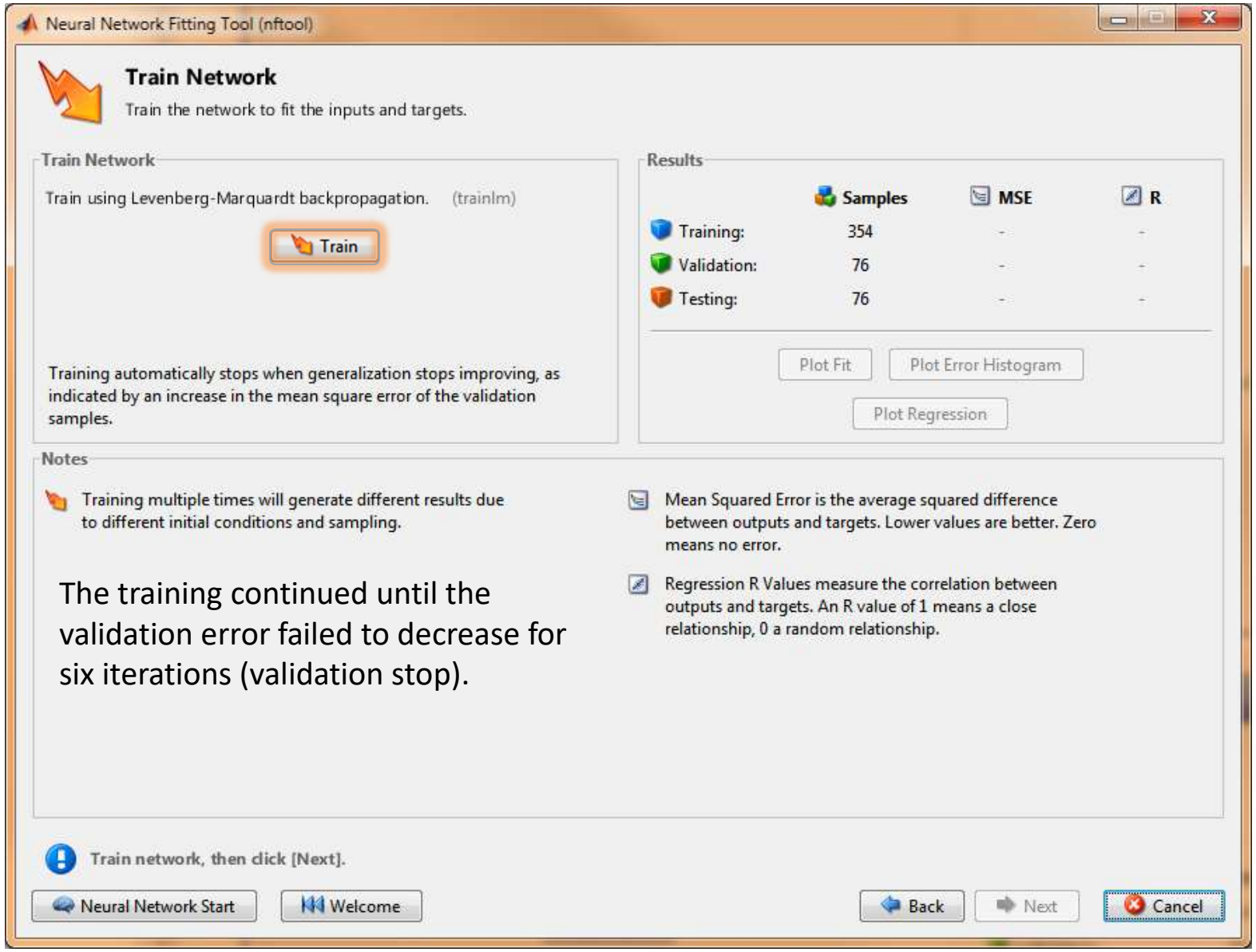
Click Next to display the Validation and Test Data window, shown in the following figure.

The validation and test data sets are each set to 15% of the original data.



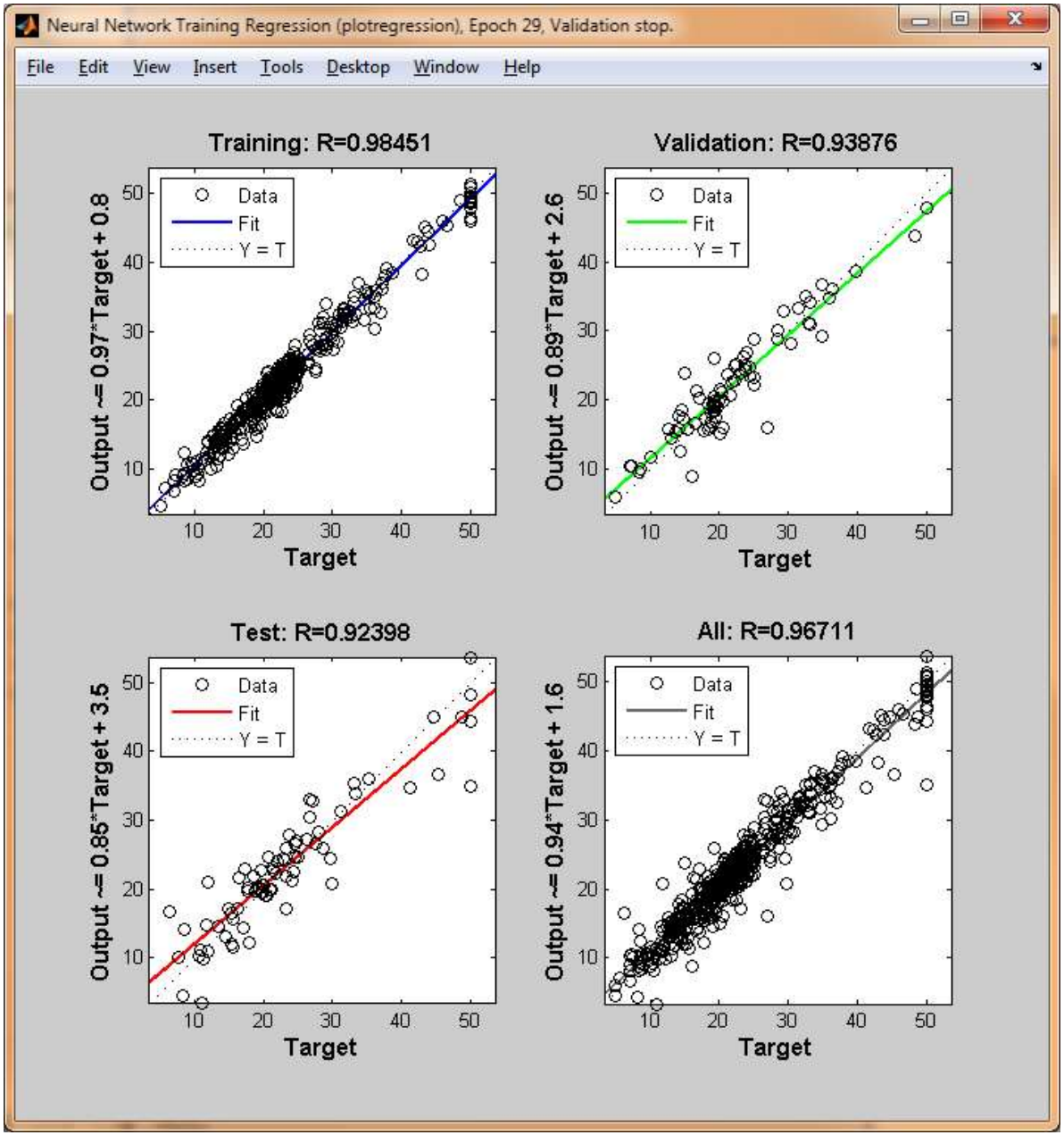
With these settings, the input vectors and target vectors will be randomly divided into three sets as follows:
70% will be used for training.
15% will be used to validate that the network is generalizing and to stop training before overfitting.
The last 15% will be used as a completely independent test of network generalization.


Neural Network Fitting Tool (nftool)
Network Architecture
Set the number of neurons in the fitting network's hidden layer.
Hidden Layer
Define a fitting neural network. (fitnet)
Number of Hidden Neurons:
10
Recommendation
Return to this panel and change the number of neurons if the network does not perform well after training.
The standard network that is used for function fitting is a two-layer feedforward network, with a sigmoid transfer function in the hidden layer and a linear transfer function in the output layer. The default number of hidden neurons is set to 10
Neural Network
Input
13
Hidden Layer
W
b
10
Output Layer
W
b
1
Output
1
Change settings if desired, then click [Next] to continue.
Neural Network Start
Welcome
Back
Next
Cancel

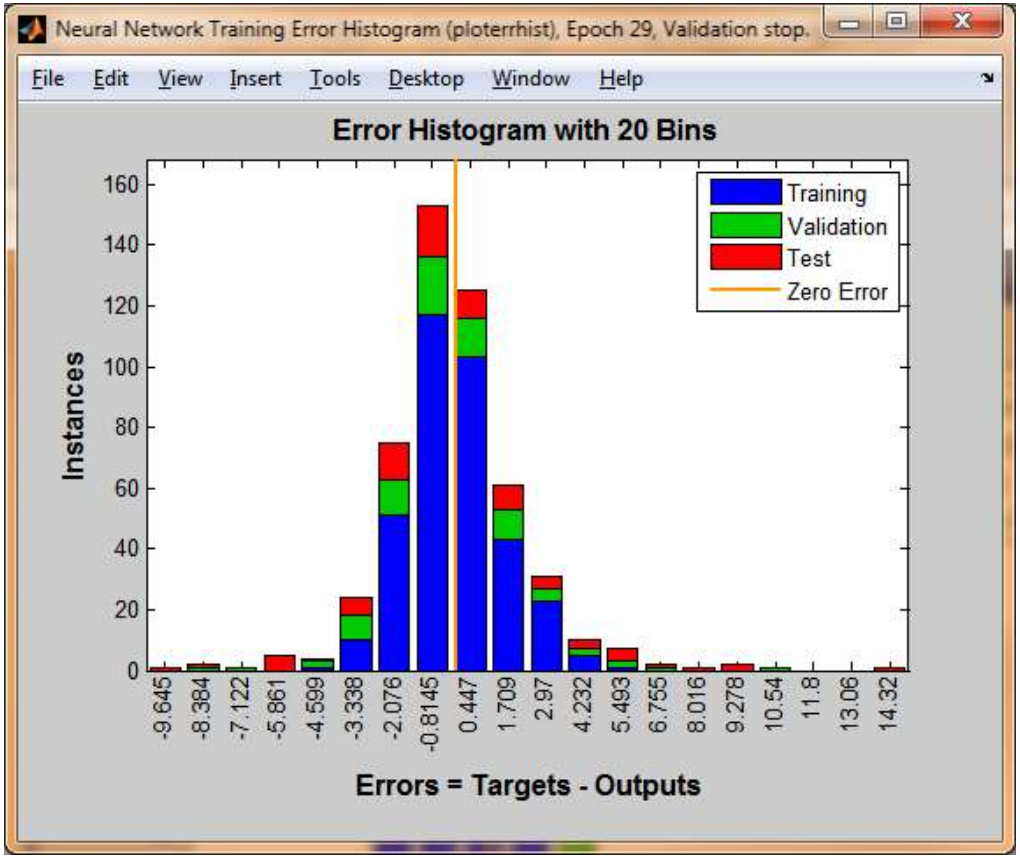


The training continued until the validation error failed to decrease for six iterations (validation stop).

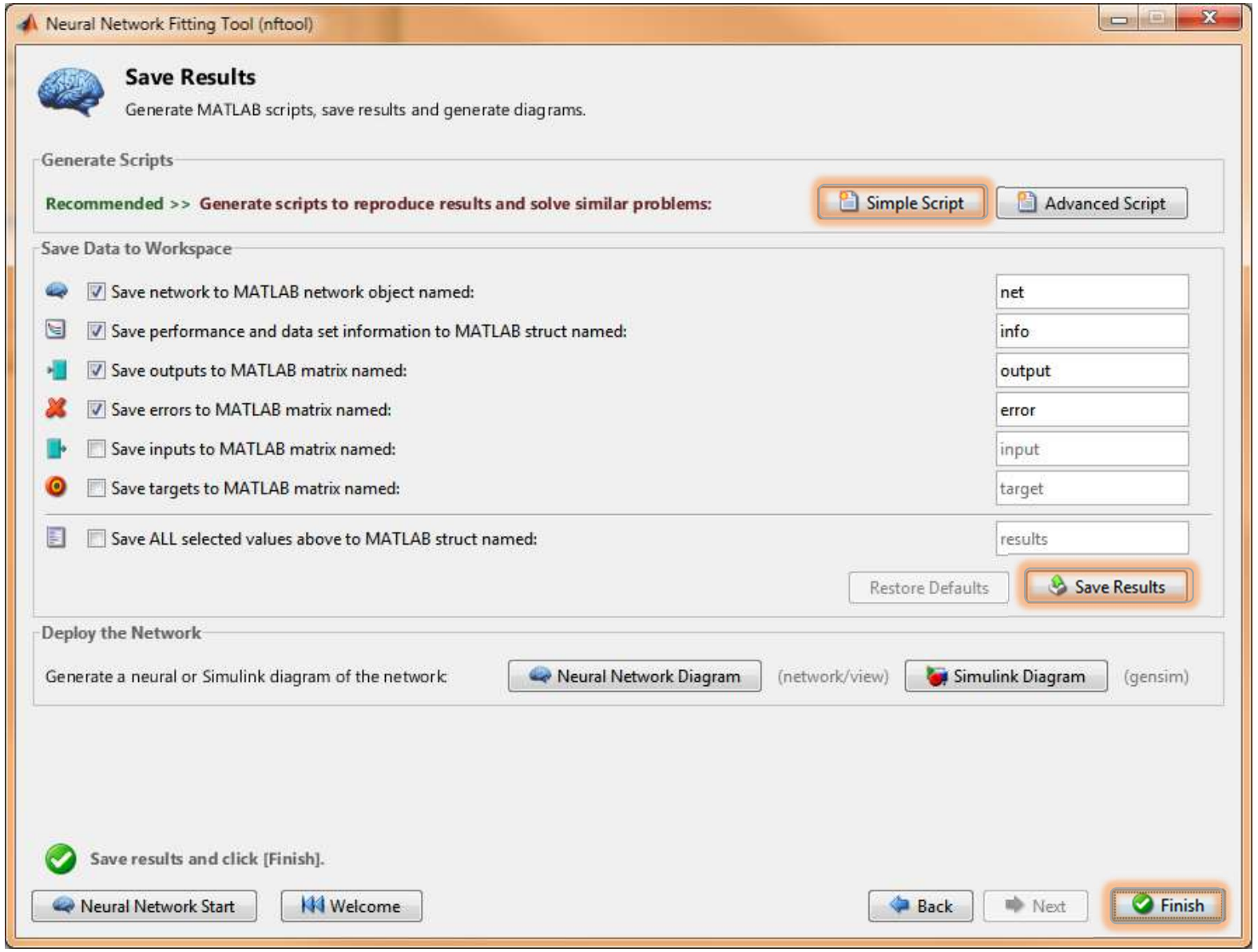
Under Plots, click Regression. This is used to validate the network performance.

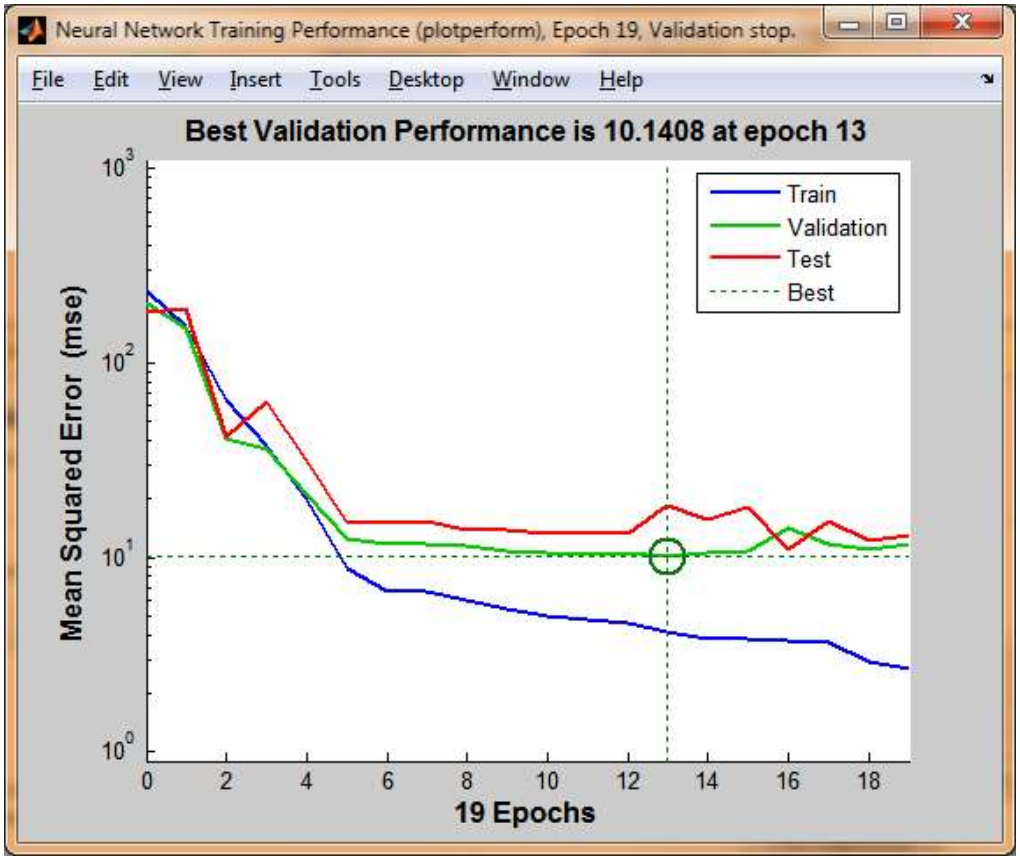


The regression plots display the network outputs with respect to targets for training, validation, and test sets. For a perfect fit, the data should fall along a 45 degree line, where the network outputs are equal to the targets. Here, the fit is reasonably good for all data sets, with R values in each case of 0.92 or above. If even more accurate results were required, you could retrain the network by clicking Retrain in nftool. This will change the initial weights and biases of the network, and may produce an improved network after retraining. Other options are provided on the following pane.



View the error histogram to obtain additional verification of network performance. Under the Plots pane, click Error Histogram.


Neural Network Fitting Tool (nftool)
Save Results
Generate MATLAB scripts, save results and generate diagrams.
Generate Scripts
Recommended >> Generate scripts to reproduce results and solve similar problems:
Simple Script
Advanced Script
Save Data to Workspace
Save network to MATLAB network object named:
net
Save performance and data set information to MATLAB struct named:
info
Save outputs to MATLAB matrix named:
output
Save errors to MATLAB matrix named:
error
Save inputs to MATLAB matrix named:
input
Save targets to MATLAB matrix named:
target
Save ALL selected values above to MATLAB struct named:
results
Restore Defaults
Save Results
Deploy the Network
Generate a neural or Simulink diagram of the network:
Neural Network Diagram
(network/view)
Simulink Diagram
(gensim)
Save results and click [Finish].
Neural Network Start
Welcome
Back
Next
Finish

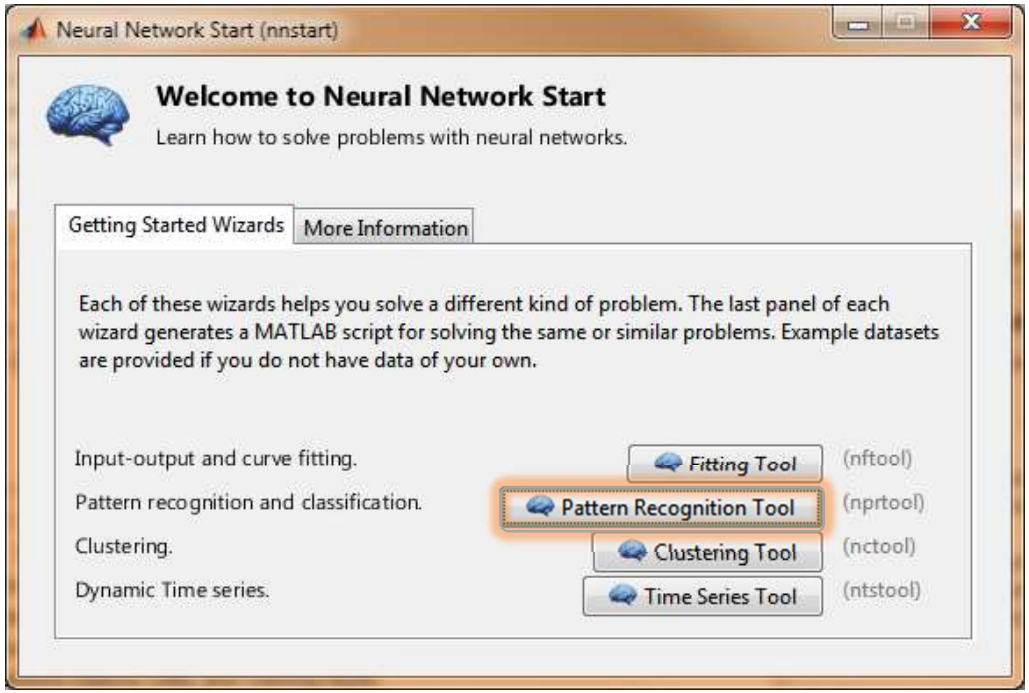

Neural Network Training Performance (plotperform), Epoch 19, Validation stop.
File Edit View Insert Tools Desktop Window Help
Best Validation Performance is 10.1408 at epoch 13
Train
Validation
Test
Best
Mean Squared Error (mse)
19 Epochs

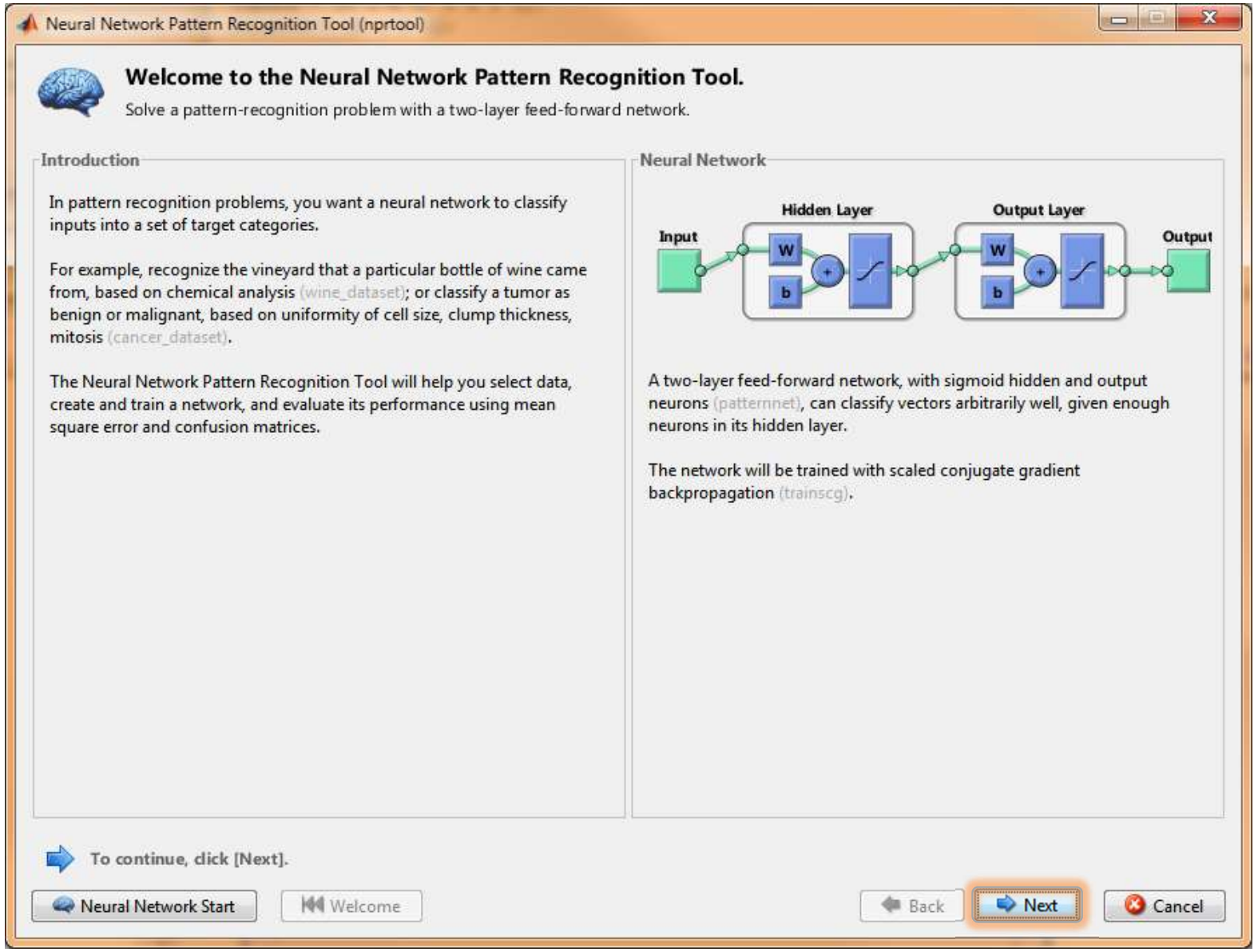
Recognizing Patterns

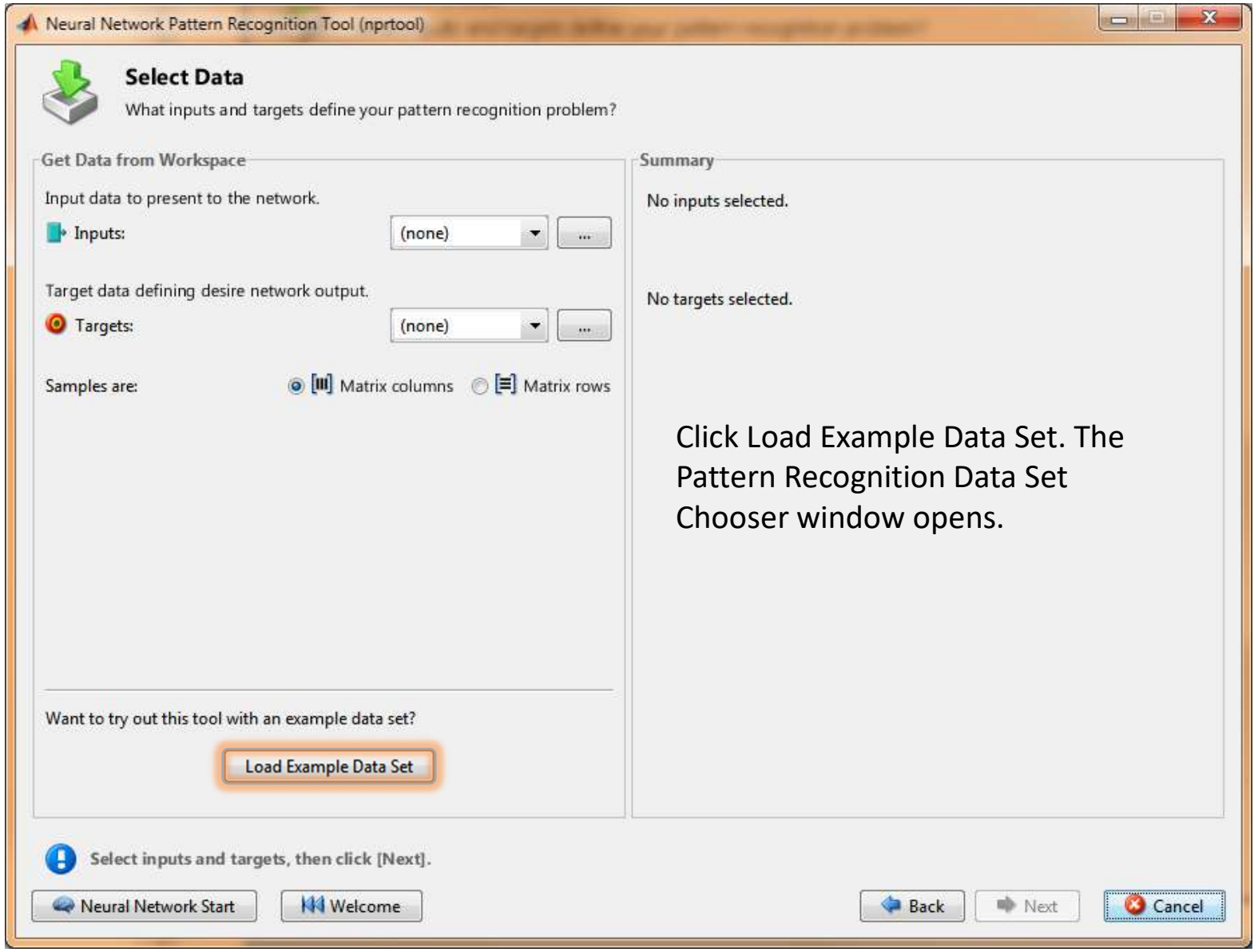
Problem Definition

- In addition to function fitting, neural networks are also good at recognizing patterns.
- For example, suppose you want to classify a tumor as benign or malignant, based on uniformity of cell size, clump thickness, mitosis, etc. You have 699 example cases for which you have 9 items of data and the correct classification as benign or malignant.
- Nnstart and click or nprtool

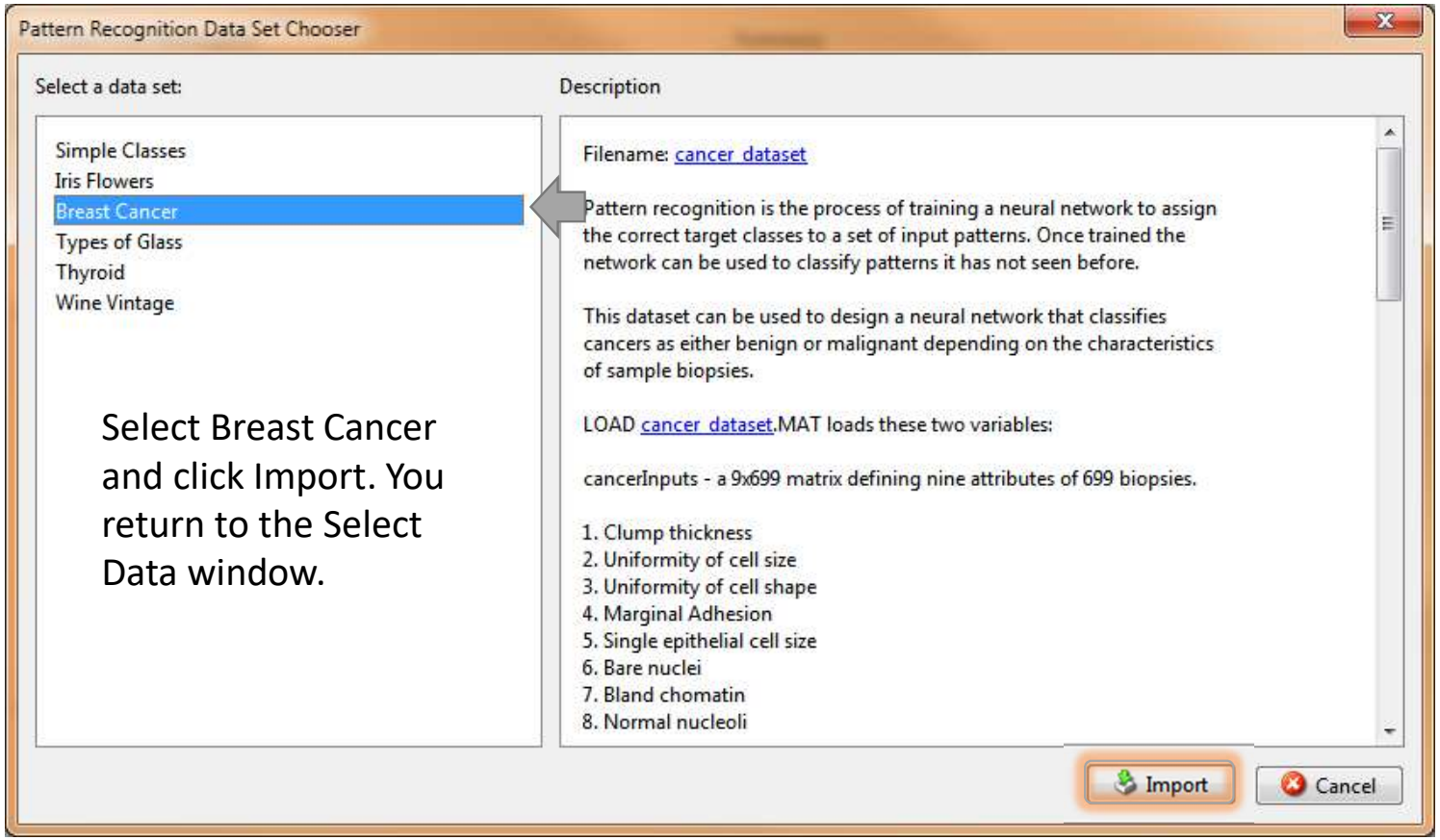
nnstart


Neural Network Start (nnstart)
Welcome to Neural Network Start
Learn how to solve problems with neural networks.
Getting Started Wizards
More Information
Each of these wizards helps you solve a different kind of problem. The last panel of each wizard generates a MATLAB script for solving the same or similar problems. Example datasets are provided if you do not have data of your own.
Input-output and curve fitting.
Fitting Tool
(nftool)
Pattern recognition and classification.
Pattern Recognition Tool
(nprtool)
Clustering.
Clustering Tool
(nctool)
Dynamic Time series.
Time Series Tool
(ntstool)

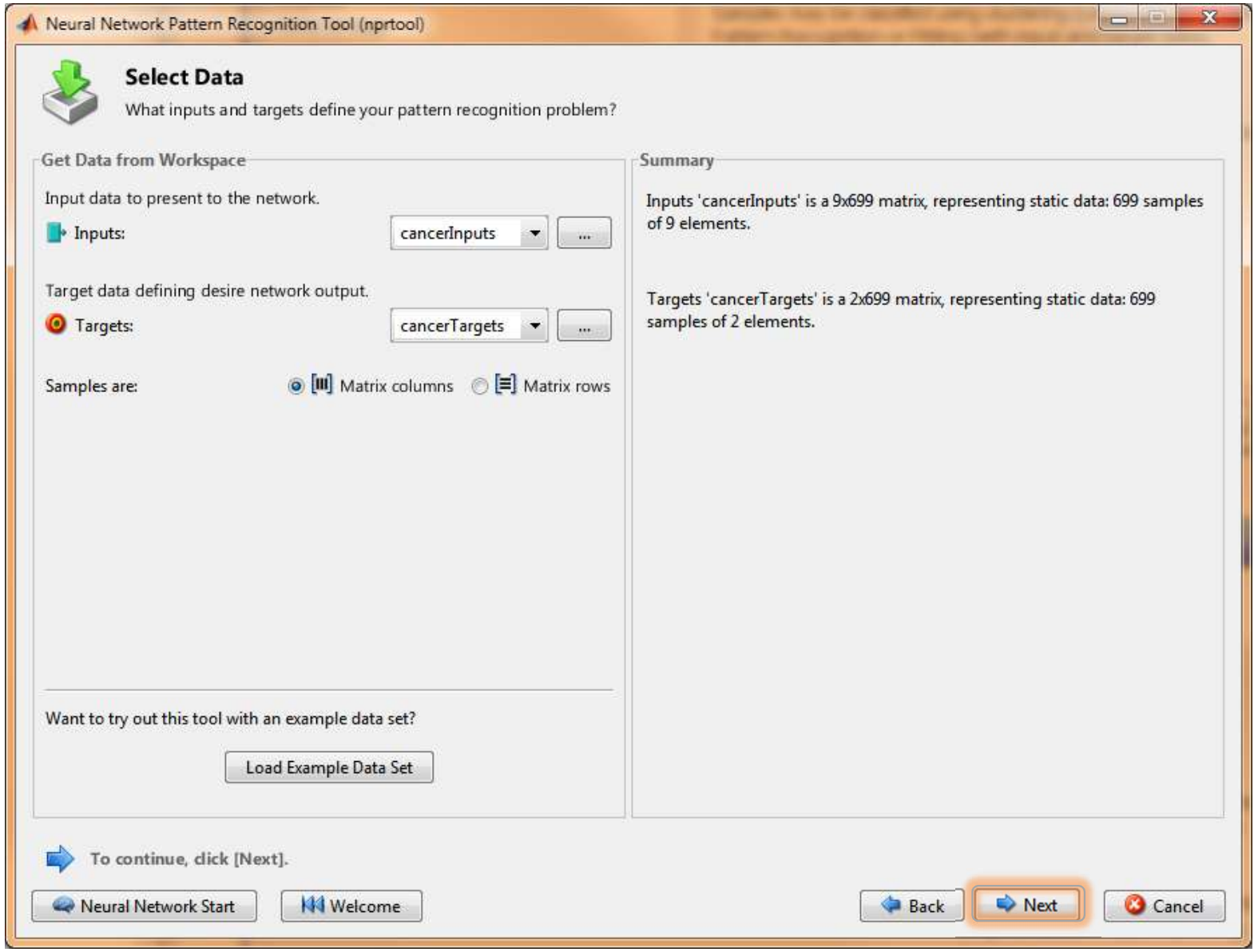

Neural Network Pattern Recognition Tool (nprtool)
Welcome to the Neural Network Pattern Recognition Tool.
Solve a pattern-recognition problem with a two-layer feed-forward network.
Introduction
In pattern recognition problems, you want a neural network to classify inputs into a set of target categories.
For example, recognize the vineyard that a particular bottle of wine came from, based on chemical analysis (wine_dataset); or classify a tumor as benign or malignant, based on uniformity of cell size, clump thickness, mitosis (cancer_dataset).
The Neural Network Pattern Recognition Tool will help you select data, create and train a network, and evaluate its performance using mean square error and confusion matrices.
Neural Network
Input
Hidden Layer
Output Layer
Output
W
b
W
b
A two-layer feed-forward network, with sigmoid hidden and output neurons (patternnet), can classify vectors arbitrarily well, given enough neurons in its hidden layer.
The network will be trained with scaled conjugate gradient backpropagation (trainscg).
To continue, click [Next].
Neural Network Start
Welcome
Back
Next
Cancel

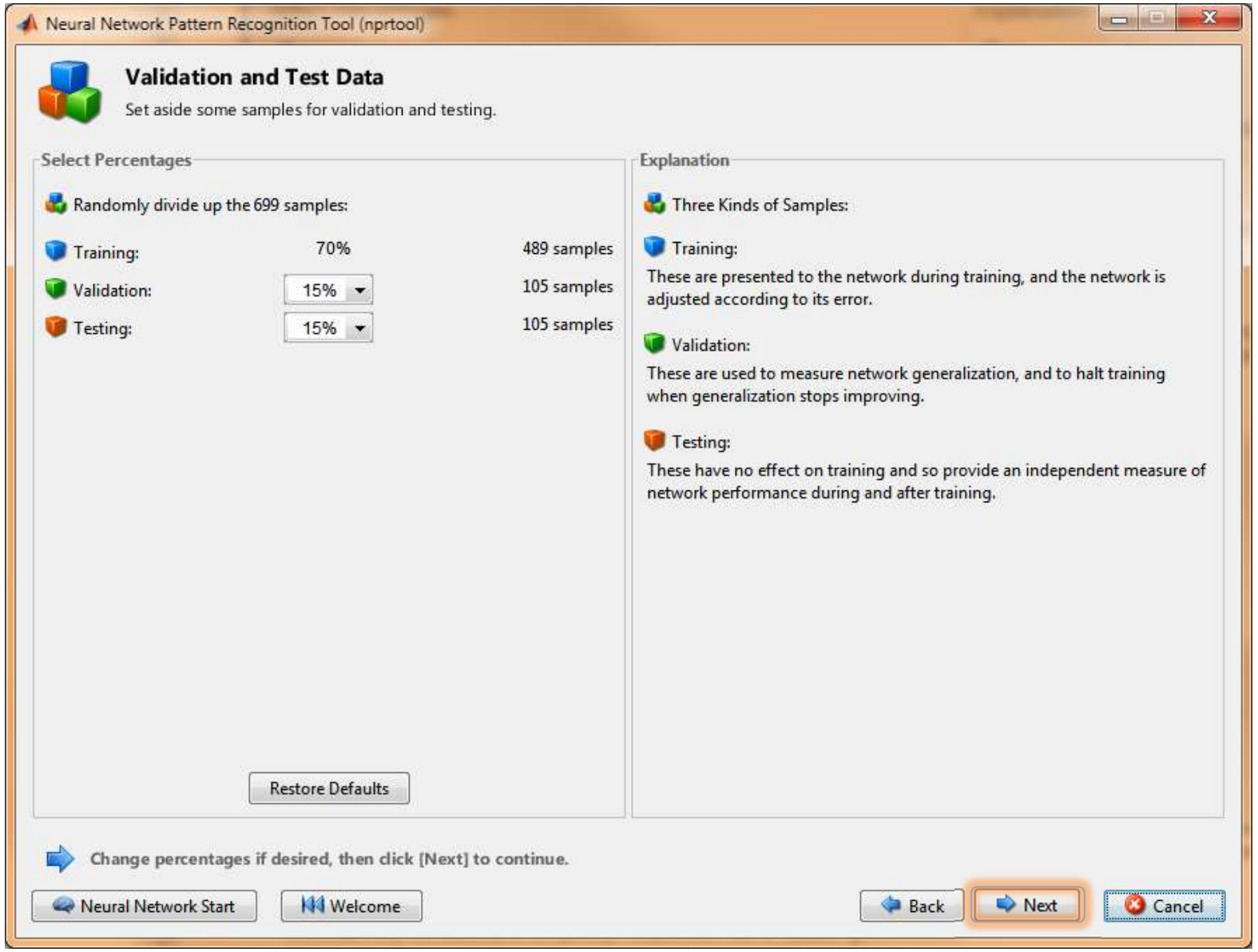


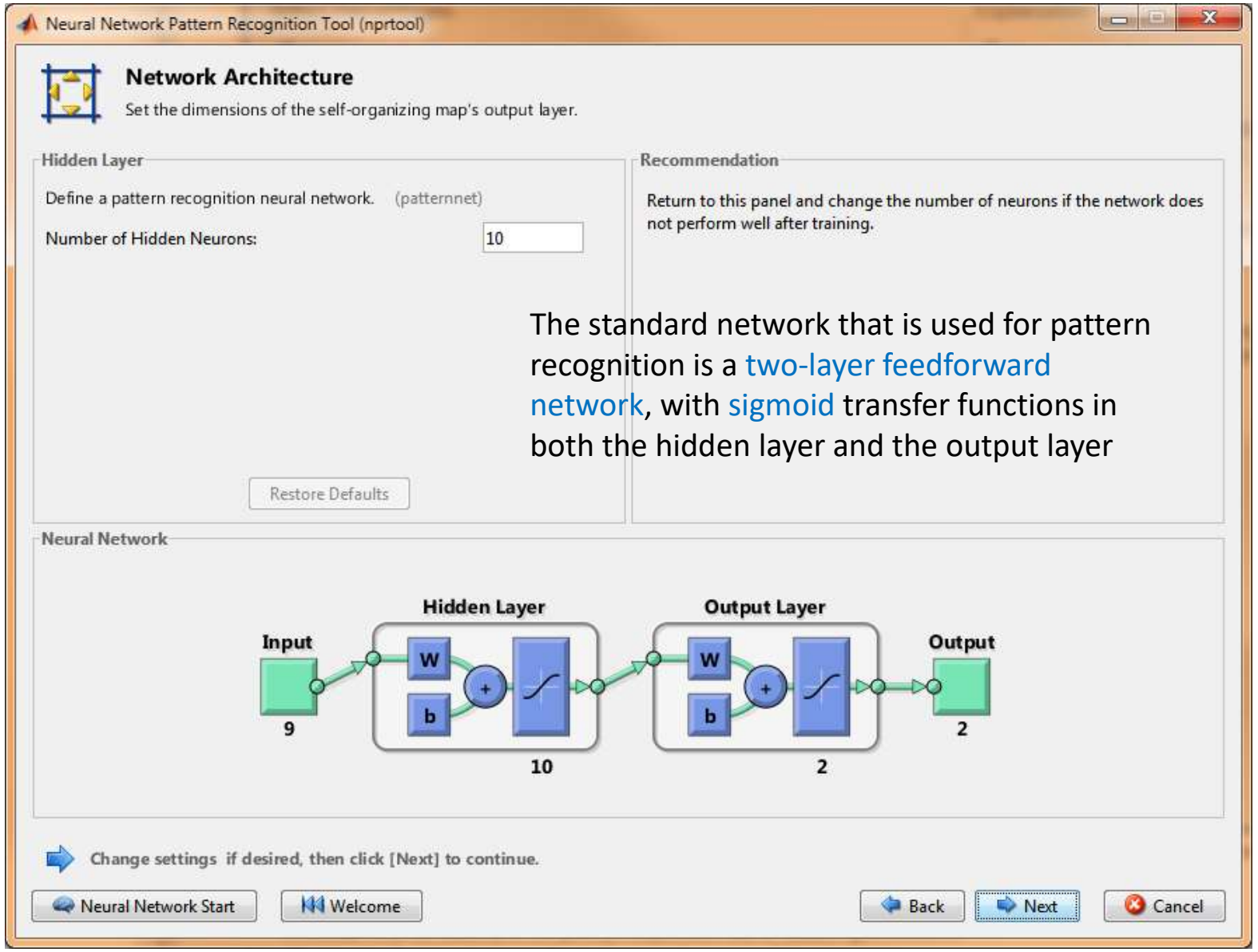
Click Load Example Data Set. The Pattern Recognition Data Set Chooser window opens.



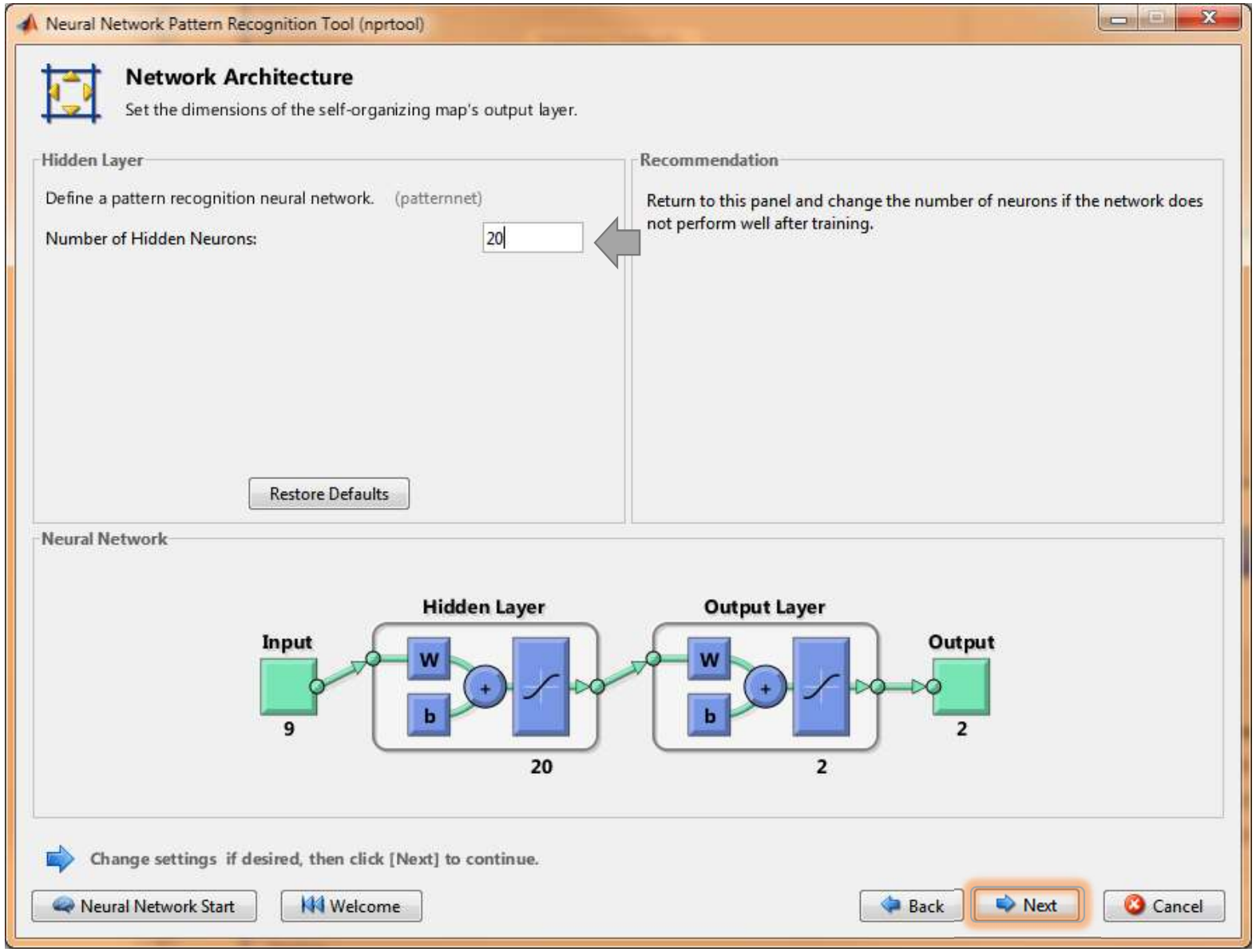
Select Breast Cancer and click Import. You return to the Select Data window.

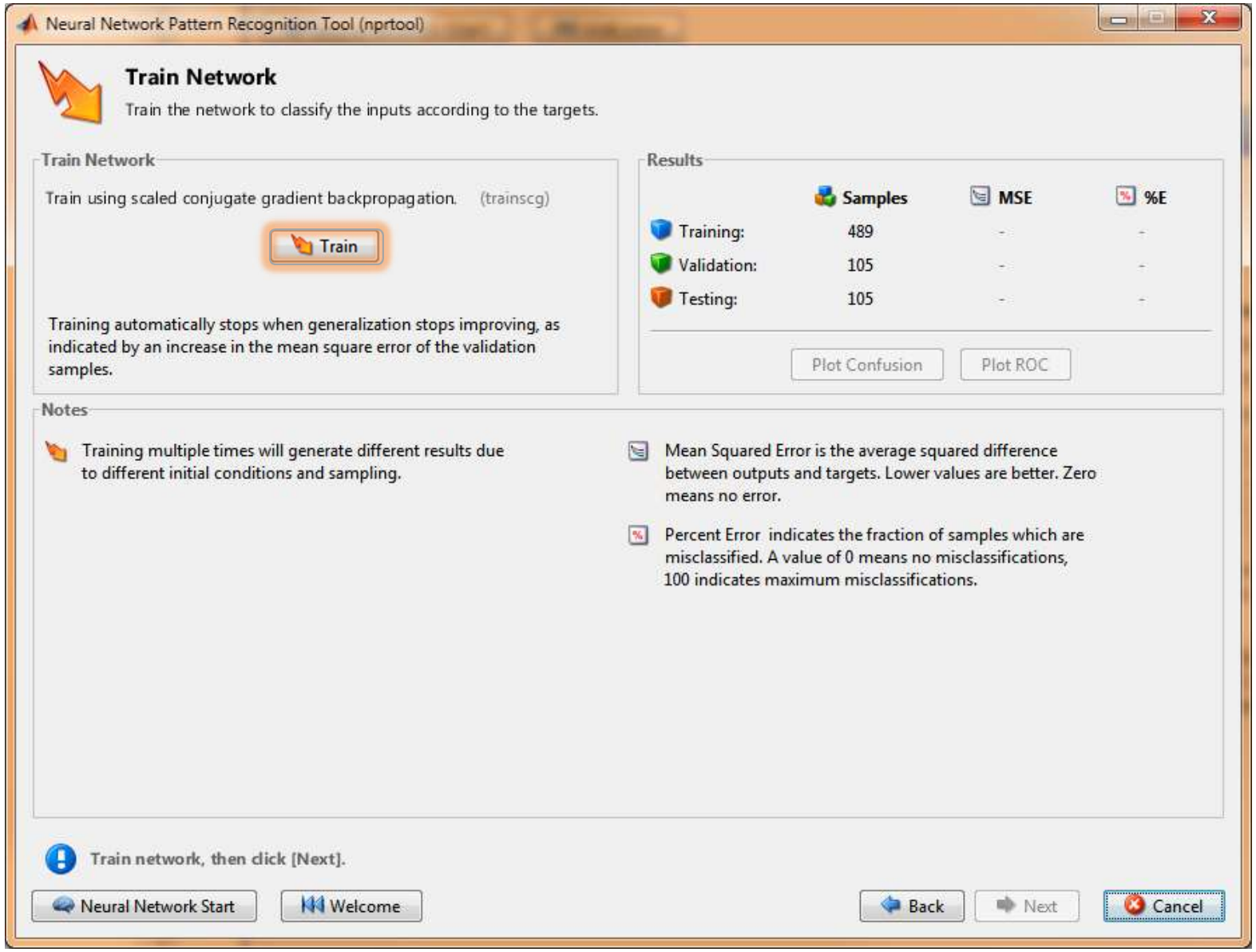

Neural Network Pattern Recognition Tool (nprtool)
Select Data
What inputs and targets define your pattern recognition problem?
Get Data from Workspace
Input data to present to the network.
Inputs:
cancerInputs
...
Target data defining desire network output.
Targets:
cancerTargets
...
Samples are:
Matrix columns
Matrix rows
Want to try out this tool with an example data set?
Load Example Data Set
Summary
Inputs 'cancerInputs' is a 9x699 matrix, representing static data: 699 samples of 9 elements.
Targets 'cancerTargets' is a 2x699 matrix, representing static data: 699 samples of 2 elements.
To continue, click [Next].
Neural Network Start
Welcome
Back
Next
Cancel

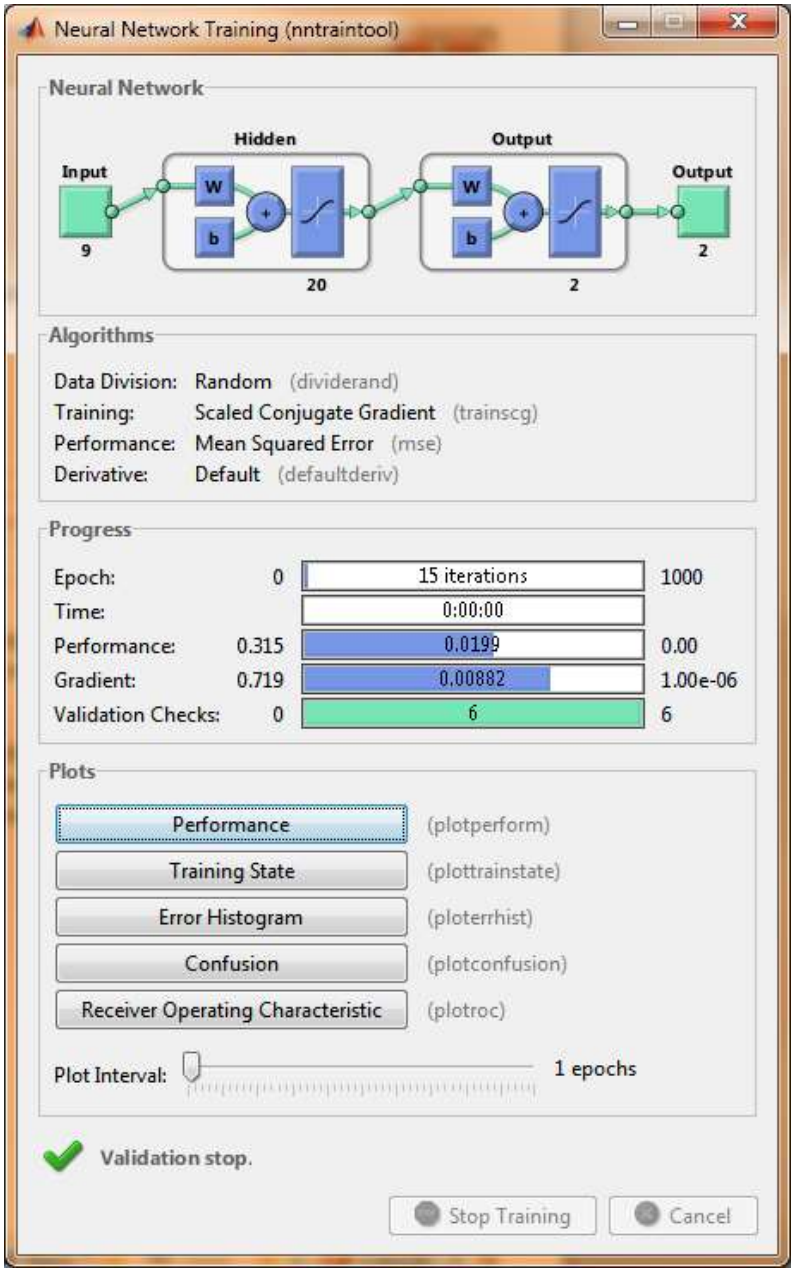

Neural Network Pattern Recognition Tool (nprtool)
Validation and Test Data
Set aside some samples for validation and testing.
Select Percentages
Randomly divide up the 699 samples:
Training: 70% 489 samples
Validation: 15% 105 samples
Testing: 15% 105 samples
Restore Defaults
Explanation
Three Kinds of Samples:
Training:
These are presented to the network during training, and the network is adjusted according to its error.
Validation:
These are used to measure network generalization, and to halt training when generalization stops improving.
Testing:
These have no effect on training and so provide an independent measure of network performance during and after training.
Change percentages if desired, then click [Next] to continue.
Neural Network Start
Welcome
Back
Next
Cancel



The standard network that is used for pattern recognition is a two-layer feedforward network, with sigmoid transfer functions in both the hidden layer and the output layer


Neural Network Pattern Recognition Tool (nprtool)
Network Architecture
Set the dimensions of the self-organizing map's output layer.
Hidden Layer
Define a pattern recognition neural network. (patternnet)
Number of Hidden Neurons:
20
Restore Defaults
Recommendation
Return to this panel and change the number of neurons if the network does not perform well after training.
Neural Network
Input
9
Hidden Layer
W
b
20
Output Layer
W
b
2
Output
2
Change settings if desired, then click [Next] to continue.
Neural Network Start
Welcome
Back
Next
Cancel

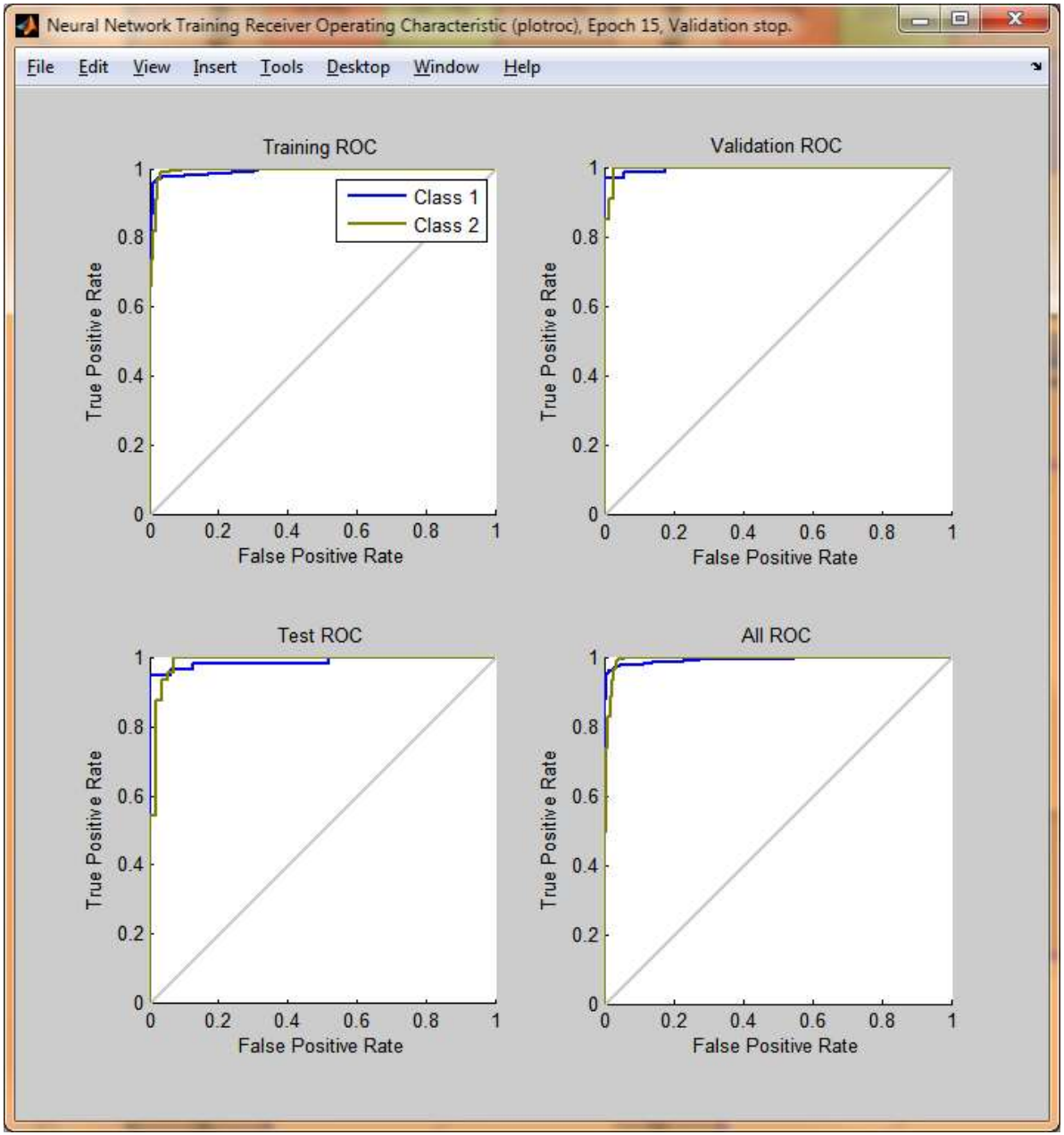

Neural Network Pattern Recognition Tool (nprtool)
Train Network
Train the network to classify the inputs according to the targets.
Train Network
Train using scaled conjugate gradient backpropagation. (trainscg)
Train
Training automatically stops when generalization stops improving, as indicated by an increase in the mean square error of the validation samples.
Results
Samples
MSE
%E
Training: 489 - -
Validation: 105 - -
Testing: 105 - -
Plot Confusion
Plot ROC
Notes
Training multiple times will generate different results due to different initial conditions and sampling.
Mean Squared Error is the average squared difference between outputs and targets. Lower values are better. Zero means no error.
Percent Error indicates the fraction of samples which are misclassified. A value of 0 means no misclassifications, 100 indicates maximum misclassifications.
Train network, then click [Next].
Neural Network Start
Welcome
Back
Next
Cancel



Under the Plots pane, click Confusion in the Neural Network Pattern Recognition Tool.

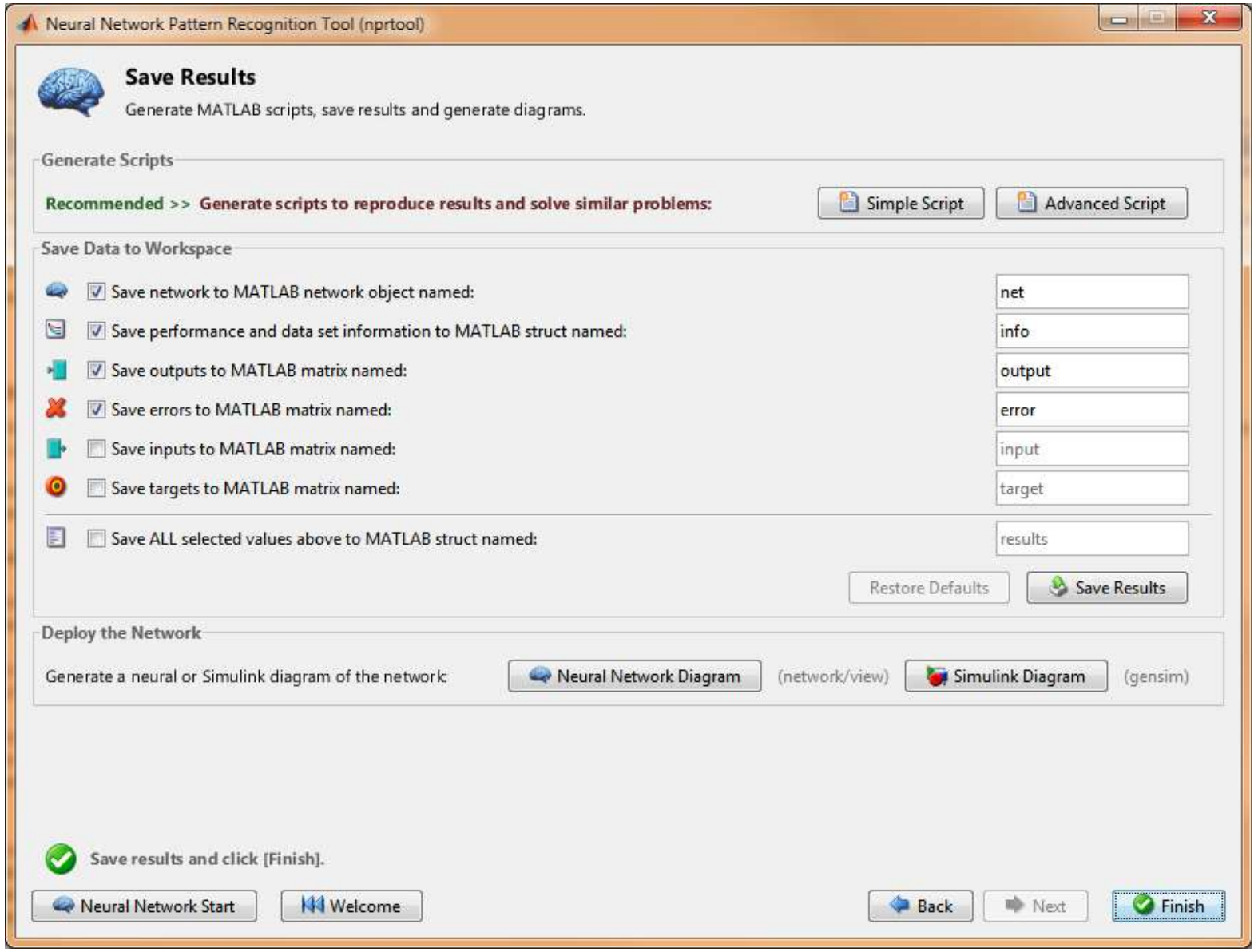
The next figure shows the confusion matrices for training, testing, and validation, and the three kinds of data combined. The network outputs are very accurate, as you can see by the high numbers of correct responses in the green squares and the low numbers of incorrect responses in the red squares. The lower right blue squares illustrate the overall accuracies.


Neural Network Training Confusion (plotconfusion), Epoch 15, Validation stop.
File Edit View Insert Tools Desktop Window Help
Training Confusion Matrix
321 65.6%
4 0.8%
98.8% 1.2%
9 1.8%
155 31.7%
94.5% 5.5%
97.3% 2.7%
97.5% 2.5%
97.3% 2.7%
Validation Confusion Matrix
69 65.7%
2 1.9%
97.2% 2.8%
2 1.9%
32 30.5%
94.1% 5.9%
97.2% 2.8%
94.1% 5.9%
96.2% 3.8%
Test Confusion Matrix
54 51.4%
2 1.9%
96.4% 3.6%
3 2.9%
46 43.8%
93.9% 6.1%
94.7% 5.3%
95.8% 4.2%
95.2% 4.8%
All Confusion Matrix
444 63.5%
8 1.1%
98.2% 1.8%
14 2.0%
233 33.3%
94.3% 5.7%
96.9% 3.1%
96.7% 3.3%
96.9% 3.1%
Output Class
Target Class
1
2



ROC Curve

The colored lines in each axis represent the ROC curves. The ROC curve is a plot of the true positive rate (sensitivity) versus the false positive rate (1 - specificity) as the threshold is varied. A perfect test would show points in the upper-left corner, with 100% sensitivity and 100% specificity. For this problem, the network performs very well.


Neural Network Pattern Recognition Tool (nprtool)
Save Results
Generate MATLAB scripts, save results and generate diagrams.
Generate Scripts
Recommended >> Generate scripts to reproduce results and solve similar problems:
Simple Script
Advanced Script
Save Data to Workspace
Save network to MATLAB network object named:
net
Save performance and data set information to MATLAB struct named:
info
Save outputs to MATLAB matrix named:
output
Save errors to MATLAB matrix named:
error
Save inputs to MATLAB matrix named:
input
Save targets to MATLAB matrix named:
target
Save ALL selected values above to MATLAB struct named:
results
Restore Defaults
Save Results
Deploy the Network
Generate a neural or Simulink diagram of the network:
Neural Network Diagram
(network/view)
Simulink Diagram
(gensim)
Save results and click [Finish].
Neural Network Start
Welcome
Back
Next
Finish