

# > New Tools for Building Predictive Models

Your organization needs to find patterns and connections in the complex and fast-changing environment you work in so that you can make better decisions at every turn. You may be using SPSS and one or more of its add-on modules to help you do this. If so, you know the power and versatility you have at your fingertips. But there's even more you can do.

You can explore subtle or hidden patterns in your data, using SPSS Neural Networks. This new add-on module offers you the ability to discover more complex relationships in your data and generate better performing predictive models. The result? Deeper insight and better decision-making.

The procedures in SPSS Neural Networks complement the more traditional statistics in SPSS Base and its modules. Find new associations in your data with SPSS Neural Networks and then confirm their significance with traditional statistical techniques.

## Why use a neural network?

A computational neural network is a set of non-linear data modeling tools consisting of input and output layers plus one or two hidden layers. The connections between neurons in each layer have associated weights, which are iteratively adjusted by the training algorithm to minimize error and provide accurate predictions. You set the conditions under which the network "learns" and can finely control the training stopping rules and network architecture, or let the procedure automatically choose the architecture for you.

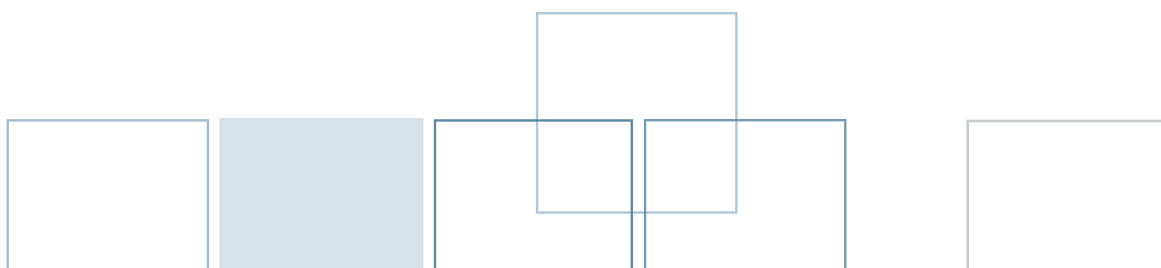
You can combine SPSS Neural Networks with other statistical procedures to gain clearer insight in a number of areas. In market research, for example, you can create customer profiles and discover customer preferences. In database marketing, you can segment your customer base and optimize marketing campaigns.

In financial analysis, you can use SPSS Neural Networks to analyze applicants' creditworthiness and to detect possible fraud. In operational analysis, use this new tool to manage cash flow and improve logistics planning. Scientific and healthcare applications include forecasting treatment costs, performing medical outcomes analysis, and predicting the length of a hospital stay.

## Control the process from start to finish

With SPSS Neural Networks, you select either the Multilayer Perceptron (MLP) or Radial Basis Function (RBF) procedure.

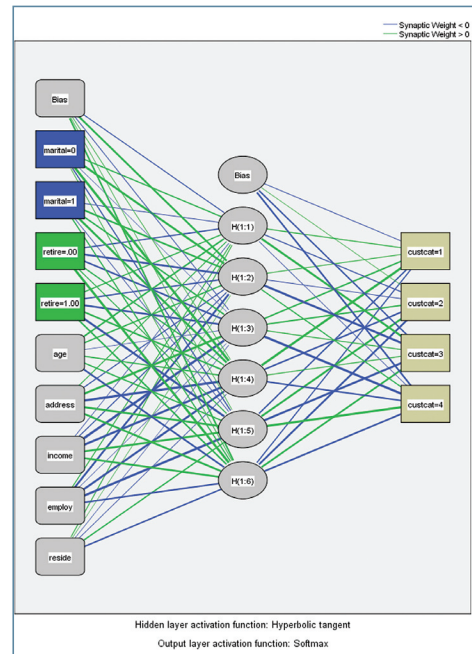
Both of these are supervised learning techniques—that is, they map relationships implied by the data. Both use feed-forward architectures, meaning that data moves in only one direction, from the input nodes through the hidden layer of nodes to the output nodes. Your choice of procedure will be influenced by the type of data you have and the level of complexity you seek to uncover. While the MLP procedure can find more complex relationships, the RBF procedure is generally faster.



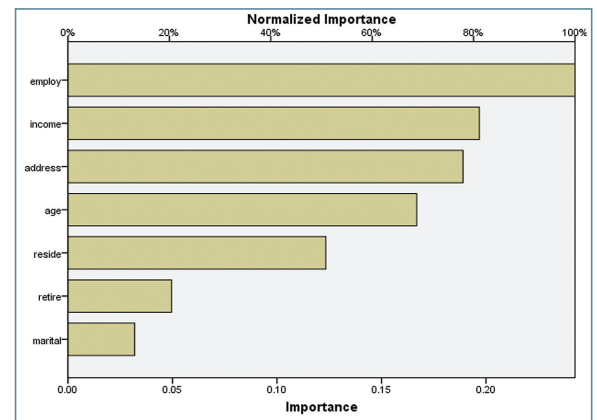
With either of these approaches, you divide your data into training, testing, and holdout sets. The training set is used to estimate the network parameters. The testing set is used to prevent overtraining. The holdout set is used to independently assess the final network, which is applied to the entire dataset and to any new data.

You specify the dependent variables, which may be scale, categorical, or a combination of the two. If a dependent variable has scale measurement level, then the neural network predicts continuous values that approximate the “true” value of some continuous function of the input data. If a dependent variable is categorical, then the neural network is used to classify cases into the “best” category based on the input predictors.

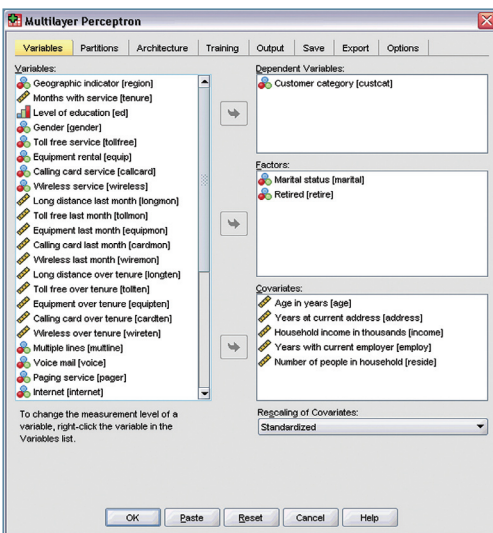
You adjust the procedure by choosing how to partition the dataset, what sort of architecture you want, and what computation resources will be applied to the analysis. Finally, you choose to display results in tables or graphs, save optional temporary variables to the active dataset, and export models in XML-file formats to score future data.



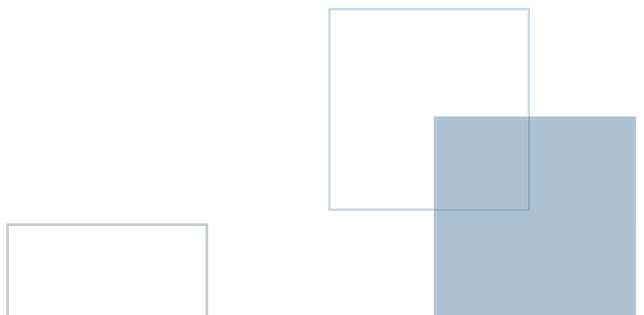
In an MLP network like the one shown here, the data feeds forward from the input layer through one or more hidden layers to the output layer.



The results of exploring data with neural network techniques can be shown in a variety of graphic formats. This simple bar chart is one of many options.



From the Multilayer Perceptron (MLP) dialog, you select the variables that you want to include in your model.

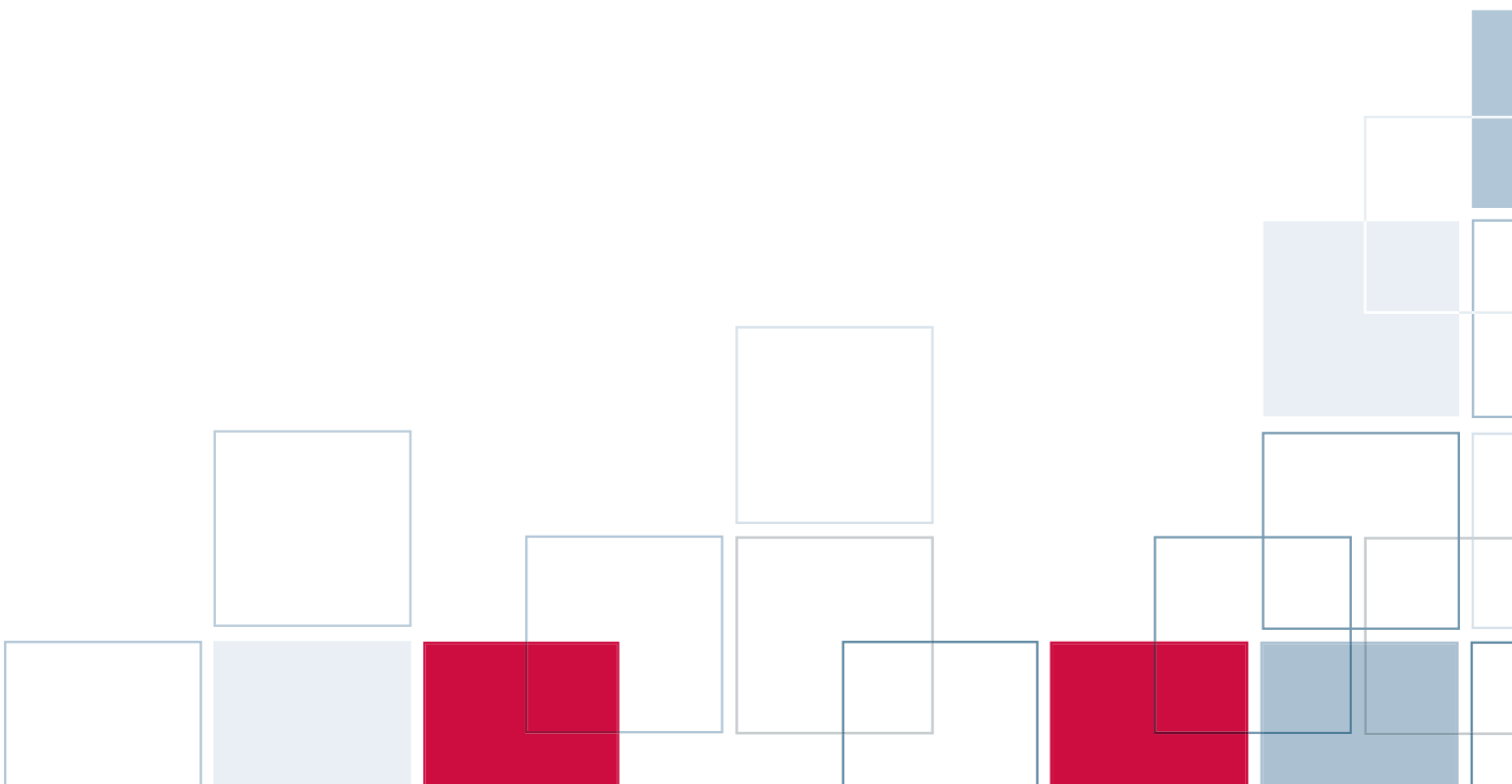


## Features

### Multilayer Perceptron (MLP)

The MLP procedure fits a particular kind of neural network called a multilayer perceptron. The multilayer perceptron is a supervised method using feedforward architecture. It can have multiple hidden layers. One or more dependent variables may be specified, which may be scale, categorical, or a combination. If a dependent variable has scale measurement level, then the neural network predicts continuous values that approximate the “true” value of some continuous function of the input data. If a dependent variable is categorical, then the neural network is used to classify cases into the “best” category based on the input predictors.

- Predictors
  - Factors
  - Covariates
- The EXCEPT subcommand lists any variables that the MLP procedure should exclude from the factor or covariate lists on the command line. This subcommand is useful if the factor or covariate lists contain a large number of variables.
- The RESCALE subcommand is used to rescale covariates or scale dependent variables
  - Dependent variable (if scale): standardized, normalized, adjusted normalized, or none
  - Covariates: standardized, normalized, adjusted normalized, or none
- The PARTITION subcommand specifies the method of partitioning the active dataset into training, testing, and holdout samples. The training sample comprises the data records used to train the neural network. The testing sample is an independent set of data records used to track prediction error during training in order to prevent overtraining. The holdout sample is another independent set of data records used to assess the final neural network. You can specify:
  - The relative number of cases in the active dataset to randomly assign to the training sample
  - The relative number of cases in the active dataset to randomly assign to the testing sample
  - The relative number of cases in the active dataset to randomly assign to the holdout sample
  - A variable that assigns each case in the active dataset to the training, testing, or holdout sample
- The ARCHITECTURE subcommand is used to specify the neural network architecture. You can specify:
  - Whether to use the automatic architecture or, if automatic is not used:
  - The number of hidden layers in the neural network
  - The activation function to use for all units in the hidden layers (Hyperbolic tangent or Sigmoid)
  - The activation function to use for all units in the output layer (Identity, Hyperbolic tangent, Sigmoid, or Softmax)
- The CRITERIA subcommand specifies the computational and resource settings for the MLP procedure. You can specify the training type, which determines how the neural network processes training data records: batch training, online training, mini-batch training. You can also specify:
  - The number of training records per mini-batch (if selected as the training method)
  - The maximum number of cases to store in memory when automatic architecture selection and/or mini-batch training is in effect
  - The optimization algorithm used to determine the synaptic weights: Gradient descent, Scaled conjugate gradient
  - The initial learning rate for the gradient descent optimization algorithm
  - The lower boundary for the learning rate when gradient descent is used with online or mini-batch training
  - The momentum rate for the gradient descent optimization algorithm
  - The initial lambda, for the scaled conjugate gradient optimization algorithm
  - The initial sigma, for the scaled conjugate gradient optimization algorithm
  - The interval [a0–a, a0+a] in which weight vectors are randomly generated when simulated annealing is used



- The STOPPINGRULES subcommand specifies the rules that determine when to stop training the neural network. You can specify:
  - The number of steps n to allow before checking for a decrease in prediction error
  - Whether the training timer is turned on or off and the maximum training time
  - The maximum number of epochs allowed
  - The relative change in training error criterion
  - The training error ratio criterion
- The MISSING subcommand is used to control whether user-missing values for categorical variables—that is, factors and categorical dependent variables—are treated as valid values
- The PRINT subcommand indicates the tabular output to display and can be used to request a sensitivity analysis. You can choose to display:
  - The case processing summary table
  - Information about the neural network, including the dependent variables, number of input and output units, number of hidden layers and units, and activation functions
  - A summary of the neural network results, including the average overall error, the stopping rule used to stop training and the training time
  - A classification table for each categorical dependent variable
  - The synaptic weights; that is, the coefficient estimates, from layer i–1 unit j to layer i unit k
  - A sensitivity analysis, which computes the importance of each predictor in determining the neural network
- The PLOT subcommand indicates the chart output to display. You can display:
  - Network diagram
  - A predicted by observed value chart for each dependent variable
  - A residual by predicted value chart for each scale dependent variable
  - ROC (Receiver Operating Characteristic) curves for each categorical dependent variable. It also displays a table giving the area under each curve.
  - Cumulative gains charts for each categorical dependent variable
  - Lift charts for each categorical dependent variable
- The SAVE subcommand writes optional temporary variables to the active dataset. You can save:
  - Predicted value or category
  - Predicted pseudo-probability
- The OUTFILE subcommand saves XML-format files containing the synaptic weights

#### Radial Basis Function (RBF)

The RBF procedure fits a radial basis function neural network, which is a feedforward, supervised learning network with an input layer, a hidden layer called the radial basis function layer, and an output layer. The hidden layer transforms the input vectors into radial basis functions. Like the MLP procedure, the RBF procedure performs prediction and classification.

The RBF procedure trains the network in two stages:

1. The procedure determines the radial basis functions using clustering methods. The center and width of each radial basis function are determined.
2. The procedure estimates the synaptic weights given the radial basis functions. The sum-of-squares error function with identity activation function for the output layer is used for both prediction and classification. Ordinary Least Squares regression is used to minimize the sum-of-squares error.

Due to this two-stage training approach, the RBF network is in general trained much faster than MLP.

Subcommands listed for the MLP procedure perform similar functions for the RBF procedure, with the following exceptions:

- When using the ARCHITECTURE subcommand, users can specify the Gaussian radial basis function used in the hidden layer: either Normalized RBF or Ordinary RBF
- When using the CRITERIA subcommand, users can specify the computation settings for the RBF procedures, specifying how much overlap occurs among the hidden units

*Features subject to change based on final product release.*



To learn more, please visit [www.spss.com](http://www.spss.com).  
For SPSS office locations and telephone numbers, go to [www.spss.com/worldwide](http://www.spss.com/worldwide).

SPSS is a registered trademark and the other SPSS products named are trademarks of SPSS Inc. All other names are trademarks of their respective owners.  
© 2007 SPSS Inc. All rights reserved. SN165PCA4-0707

