
Article from

# Predictive Analytics and Futurism

July 2016
Issue 13

# Regression and Classification: A Deeper Look

By Jeff Heaton

Classification and regression are the two most common forms of models fitted with supervised training. When the model must choose which of several discrete classes the sample data belong to, classification is used. Similarly, when the model must compute a numeric output from the input data, regression is used. Classification is used when the output is discrete, or categorical, and regression is used when the output is continuous, or numeric.

It quickly becomes more complex than this simple case. Many models, such as support vector machines or generalized linear models (GLMs), only support binary classification—they can only directly classify between two discrete classes. Yet these models are often used for many more than two classes. Similarly, neural networks and linear regression only directly support regression. This article will look at three distinct applications of supervised learning:

- binary classification
- multi classification
- regression

The exact means by which several models support these three will be discussed. This article will specifically examine the following models:

- generalized linear regression (GLM)
- linear regression
- neural networks
- support vector machines
- tree-based models

## SIMPLE REGRESSION

Linear regression is one of the most basic, yet still useful, types of model. One representation of the linear regression formula is given by Equation 1.

**Equation 1: Linear Regression**

$$\hat{y} = \beta_1 x_1 + \cdots + \beta_n x_n$$

The output prediction $\hat{y}$ (y-hat) is calculated by the summation of multiplying each input element ($x$) by a corresponding coefficient/weight ($\beta$). Training the linear regression model is simply a matter of finding the coefficient values that minimize the difference between $\hat{y}$ and the actual y. It is very common to append a constant value, typically 1, to the input vector ($x$). This constant allows one of the coefficient ($\beta$) values to serve the y-intercept.

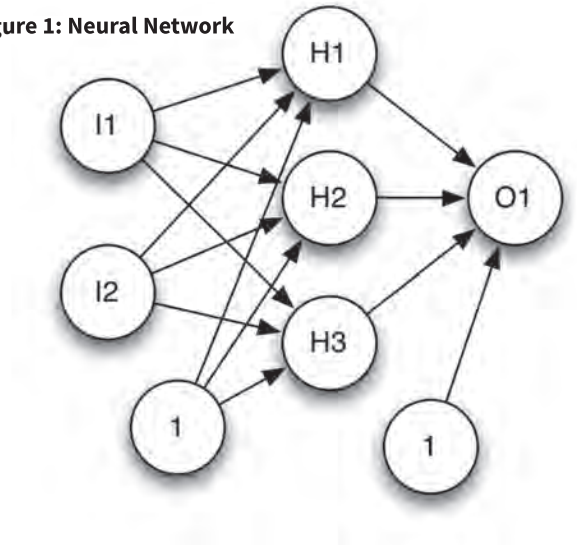
The returned value is numeric—a regression was performed. Common examples of linear regressions derive coefficients to determine shoe size, based on height, or a person's income based on several other numeric observations. Linear regression is best at modeling linear relationships. For nonlinear relationships, a neural network or generalized linear model (GLM) might be used. A single neuron in a neural network is calculated by Equation 2.

**Equation 2: GLM or Single Neuron Calculation**

$$\hat{y} = f(x, w) = \Phi(x_1 w_1 + \cdots + x_n w_n)$$

The output from a single neuron is very similar to the linear regression. An input/feature vector ($x$) is still the primary input. However, neural network terminology usually refers to the coefficients ($\beta$) as weights ($w$). Usually a constant input term is appended, just like linear regression. However, neural networks terminology refers to this weight as a bias or threshold, rather than the y-intercept. The entire summation is passed to a transfer, or activation function, denoted by $\Phi$. The transfer function is typically sigmoidal, either logistic or the hyperbolic tangent. Newer neural networks, particularly deep neural networks, will often use a rectifier linear unit (ReLU) as the transfer function. Neural networks are typically made of layers of neurons, such as Figure 1.

**Figure 1: Neural Network**



The output from a neural network is calculated by first applying the input vector ($x$) to the input neurons, in this case I1 and I2. A neural network must always have the same number of input neurons as the vector size of its training data ($x$). Next, calculate the values of each hidden neuron H1, H2, etc., working forward until the output neuron(s) are calculated.

The output for a GLM is calculated exactly the same as a single neuron for a neural network. However, the transfer/activation function is referred to as a link function. Because of this, a neural network can be thought of as layers of many GLMs.

The error for a neural network or GLM can be thought of as the difference between the predicted output ($\hat{y}$) and the expected output ($y$). A common measure of the error of neural networks, and sometimes GLMs, is root mean square error (RMSE), the calculation for which is shown by Equation 3.

**Equation 3: RMSE**

$$E = \sqrt{\frac{\sum_{i=1}^{N}(\hat{y}_t - y_t)^2}{N}}$$

The constant N represents the number of items in the training set. RMSE is very similar to the standard deviation calculation used in statistics. RMSE measures the standard deviation from the expected values.

## BINARY CLASSIFICATION

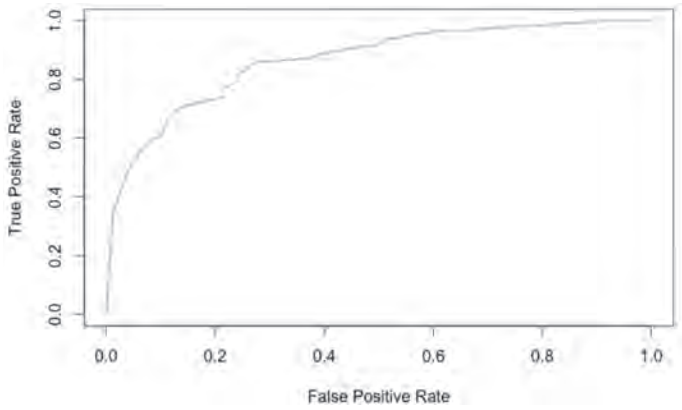
Binary classification is when a model must classify the input into one of two classes. The distinction between regression and binary classification can be fuzzy. When a model must perform a binary classification, the model output is a number that indicates the probability of one class over the other. This classification is essentially a regression on the probability of one class vs. the other being the correct outcome! For many models, binary classification is simply a special case of regression.

A popular form of binary classification for the GLM model is logistic regression, where the link function is the logistic function. If the GLM, using logistic regression, is to classify more than two categories, a special voting arrangement must be used. This is discussed later in this article.

Binary classification provides a number that states the probability of an item being a member of a category. However, this brings up the question of what a sufficient probability is for classification. Is a 90 percent probability enough? Perhaps a 75 percent probability will do. This membership threshold must be set with regard to the willingness to accept false positives and false negatives. A higher threshold decreases the likelihood of a false positive, at the expense of more false negatives. A receiver

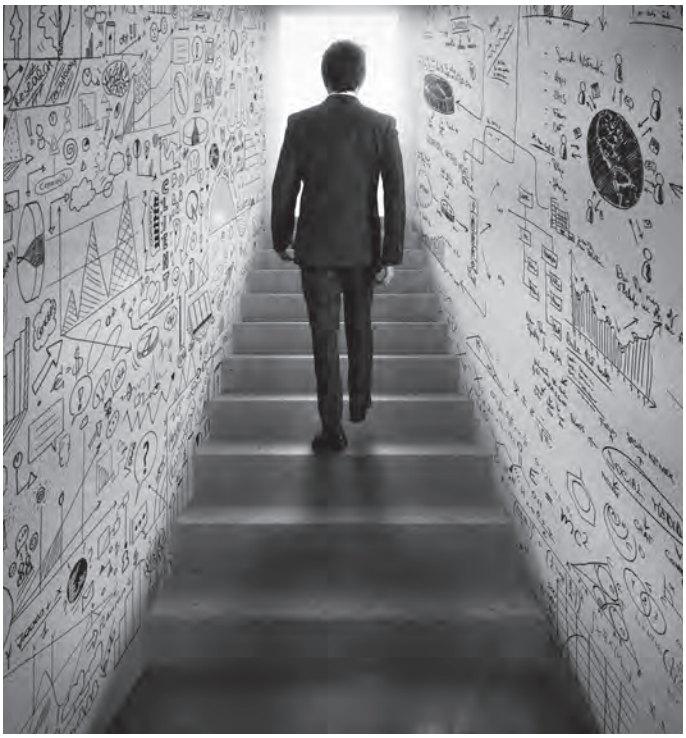
operating characteristic (ROC) curve is often used, for binary classification, to visualize the effects of setting this threshold. Figure 2 shows a ROC curve.

**Figure 2: ROC Curve**




As the threshold is set more or less restrictive, the true positive rate and false positive rates change. A threshold is represented as one point on the curved line. If the true positive rate is very high, so will be the false positive rate. The reverse also holds true.

Sometimes it is valuable to measure the effectiveness of the model, independent of the choice of threshold. For such cases, the area under the curve (AUC) is often used. The larger the



area below the curve, the better the model. An AUC of 1.0 is a perfect, but highly suspicious, model. Nearly "perfect" models are rare, and usually indicate overfitting. Calculating the AUC can be complex and often employs similar techniques to integral estimation. It is rare that AUC calculation can be performed using definite symbolic integration.

## MULTIPLE CLASSIFICATION

AUC curves are only used for binary classification. If there are more than two categories, a confusion matrix might be used. The confusion matrix allows the analyst to quickly see which categories are often mistaken for each other. A confusion matrix for the classic iris dataset is shown by Figure 3.

**Figure 3: Iris Confusion Matrix**

| | | Predicted | | |
|---|---|---|---|---|
| | | Setosa | Versicolor | Virginica |
| Actual | Setosa | 46 | 1 | 3 |
| | Versicolor | 2 | 46 | 1 |
| | Virginica | 1 | 1 | 48 |

The iris dataset is a collection of 150 iris flowers, with four measurements from each. Additionally, each iris is classified as one of three species. This dataset is often used for example classification problems. The confusion matrix shows that the model in question predicted Setosa correctly 46 times, but misclassified a Setosa as Versicolor once, and Virginica three times. A strong model will have its highest values down the northwest diagonal of a confusion matrix.

It is important to understand how a model reports the prediction for a multiclassification. A model will report a vector for the input data. The model might report 90 percent Setosa, 7 percent Versicolor, and 3 percent Virginica for a set of flower measurements that the model felt was likely Setosa. In this case, the model would return the following vector:

[0.9,0.07,0.03]

This is very different from the typical multiple choice question format that might be seen on an actuarial exam. For such an exam, the answer must be chosen as either A, B, C or D. The model has the advantage of being able to choose its confidence in each of the possible choices.

Classification problems are often numerically evaluated using the multiple log loss, as shown by Equation 4.

**Equation 4: Multi-Log Loss**

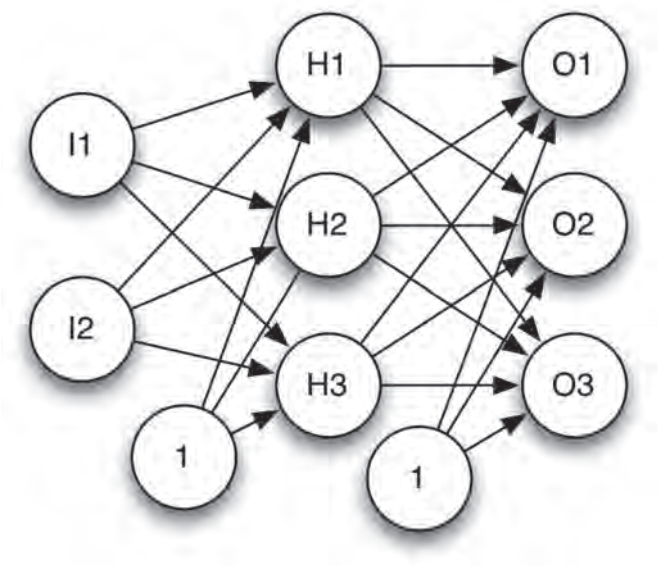
$$E = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j}\log(\hat{y}_{i,j})$$

The constant N represents the number of training set items and M represents the number of classes. Like previous equations in this article, y represents the model prediction and ŷ represents the expected outcome. The lower the log loss, the better. To explain how Equation 4 works, think of the multiple choice exam previously mentioned. If the correct answer for a particular question was A, and the model had given a .97 probability to A, then -log(0.97) points would be added to the average score. Log loss can be harsh. Predicting 1.0 correctly will add zero log-points to the error, but predicting 1.0 incorrectly will give an infinitely bad score. Because of this, most models will never predict 1.0.

## MULTIPLE REGRESSION

Just like multiple classification, multiple regression also exists. Neural networks with multiple outputs are multiple regression models. Usually, a neural network with multiple outputs is used to model multiple classification. This is how neural networks, which are inherently regressive, are made to support classification. A binary classification neural network simply uses a single output neuron to indicate the probability of the input being classified into one of the two target categories. For three or more categories, the output neurons simply indicate the class that has the greatest probability. Figure 4 shows a multiple output neural network.

**Figure 4: Multi-Output Neural Network**




Because a binary classification neural network contains a single output neuron, and a three or more classification network would contain a count equal to the number of classes, a two-output neuron neural network is rarely used. While a neural network could be trained to perform multiple regressions simultaneously, this practice is not recommended. To regress multiple values, simply fit multiple models.

## SOFTMAX CLASSIFICATION

For classification models, it is desired that the probabilities of each class sum to 1.0. Neural networks have no concept of probability. The output neuron, with the highest value, is the predicted class. It is useful to balance the output neurons of neural networks, and some other models, to mirror probability. This is accomplished with the softmax, as shown by Equation 5.

**Equation 5: Softmax**

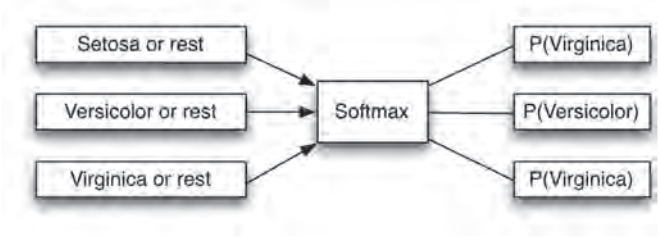
$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}}, j = 1, \ldots, K.$$

Softmax can be used to transform any multi-output regression model to a classification model. Most neural network-based classifications make use of the softmax function. It is based on the logistic function and provides the same sort of squashing effect at the extremities. The softmax function is very similar to simply summing the output neurons and balancing each neuron to become the proportion of this summation. This approach is often called normalization or simply hardmax. The softmax softens this approach and usually makes the probabilities more realistic.

## VOTING

Many models can only function as binary classifiers. Two such examples are GLMs and support vector machines (SVM). Not all models have this limitation; any tree-based model can easily classify beyond two classes. For a tree, the leaf-node specifies the class. It is very easy to convert any binary classifier into a three or more classifier. Figure 5 shows how multiple binary classifiers could be adapted to the iris dataset for classification.

**Figure 5: Model Voting**




Essentially, a classifier is trained for each of the output categories. For the iris dataset, three additional datasets are created, each as a binary classifier dataset. The first dataset would predict between Setosa and all other classes. Each class would have a dataset and model that predicts the binary classes of that category and all others. When using such a model, the input data would be presented to each of the three models, and the data would be classified as belonging to the class that predicted the highest probability.

Like a multi-output neural network, it would be helpful if the probabilities of each class summed to 1.0. This can be accomplished with the softmax function. By using multiple binary classifiers and a softmax, any binary classifier can be expanded beyond two classifications.

## CONCLUSIONS

Classification and regression are the two most common formats for supervised learning. As this article demonstrated, models have entirely different approaches to implementing classification and regression. Often the software package will take care of these differences. However, understanding the underpinnings of the models can be useful. For example, if a model were trained to recognize a large number of classes, then a GLM or SVM might not be a good choice. If there were 10,000 possible outcome classes, a binary-only classifier would need to create a voting structure of 10,000 models to vote upon each classification. A large tree/forest or neural network might be able to more effectively handle such a problem. ■

Jeff Heaton is the author of the "Artificial Intelligence for Humans" series of books, and senior data scientist at RGA Reinsurance Co. in Chesterfield, Mo. He can be reached at *jheaton@rgare.com.*