Specifying regression

This article offers some suggestions on how to specify regression models and rules for fitting regression models to data. It provides an alternative to a popular style that I find particularly confusing and bizarre, which involves presenting a model something like this:

$$y_i = \alpha + \boldsymbol{\beta X}_i + \varepsilon_i, \quad \text{where } \varepsilon_i \sim N(\mu, \sigma^2)$$

What is wrong with this notation?

In this notation the idea is that all the terms represent numbers, not distributions or anything more complex. The last term, ε_i, is special because it represents the numbers that make each equation balance and are also in some way distributed Normally.

In this style a general rule is presented almost as if it were just one example. The fact that this is a general rule is conveyed only by the suffixes and, concerning those, some important information is unstated. What does i represent? Is this equation true for all the data we already have (and will use to train the model), or is it true for all imaginable cases, or some other set? Something like "$\forall\, i : \mathbb{N} \mid i \in \cdots$" would do the trick but there are other ways, often better.

Despite the claim that $\varepsilon_i \sim N(\mu, \sigma^2)$, the reality is that ε_i represents a vector of numbers, one for each equation. A vector of numbers like this is not a distribution, though they may look like numbers that might have been selected by generating random numbers according to the Normal distribution. A set of numbers that could have been drawn like this is not the same, logically, as a probability density function and some vectors might be more typical of draws from a Normal distribution than others, yet still plausible, or at least possible.

It also seems inherently confusing to try to express information about the model and an assumption about the data in one statement.

Sometimes it is not clear whether the model will predict a specific value for y or the probability that any given value for y is seen, given the value of X.

Finally, the rule for 'fitting' the model to data is not specified and often is crucial to the results achieved.

An alternative style

A clear and correct way to specify regression should clearly separate information about the model, assumptions about the situation, and the method of fitting. It should do this without any unspecified or clashing types.

Specifying models

For example, to define a model called m as the linear model stated above (and assuming 5 predictor variables and one variable to be predicted) we can write something like:

$m : \mathbb{R}^5 \to \mathbb{R}$

$\alpha : \mathbb{R}$

$\boldsymbol{\beta} : \mathbb{R}^5$

$\forall \boldsymbol{X} : \mathbb{R}^5 \mid \boldsymbol{X} \in \mathbb{R}^5 \cdot m[\boldsymbol{X}] = \alpha + \boldsymbol{\beta X}$

where $\boldsymbol{X}$ is an illustrative vector of five Real numbers representing the values of five continuous predictor variables. This model takes as input a vector of five variable values and returns a single number that is its best guess for y.

Notice how the mathematical type of each object is given before it is used.

(In the line containing $m[\boldsymbol{X}] = \alpha + \boldsymbol{\beta X}$, the juxtaposition of $\boldsymbol{\beta}$ and $\boldsymbol{X}$ means something different from simple multiplication of numbers. Since these are both vectors, the operation is assumed to be a form of vector multiplication that ends with a single number as the result that can then be added to α. This is completely normal in mathematics and not a special feature of this style of definition. We don't usually remark on it.)

A more sophisticated model that provides a probability distribution rather than just a best guess could instead use:

$m : \mathbb{R}^5 \to (\mathbb{R} \to \mathbb{R})$

$\alpha : \mathbb{R}$

$\boldsymbol{\beta} : \mathbb{R}^5$

$\sigma : \mathbb{R}$

$\forall \boldsymbol{X} : \mathbb{R}^5 \mid \boldsymbol{X} \in \mathbb{R}^5 \cdot m[\boldsymbol{X}] = N[\alpha + \boldsymbol{\beta X}, \sigma^2]$

where N is a standard function that gives a Normal distribution when supplied with values for the mean and standard deviation. All that is left is to decide particular values for α, $\boldsymbol{\beta}$, and σ.

An equivalent statement of this model would replace the final line with:

$\forall \boldsymbol{X} : \mathbb{R}^5 \mid \boldsymbol{X} \in \mathbb{R}^5 \cdot m[\boldsymbol{X}] = \alpha + \boldsymbol{\beta X} + N[0, \sigma^2]$.

(In the line $m[\boldsymbol{X}] = \alpha + \boldsymbol{\beta X} + N[0, \sigma^2]$ the juxtaposition of $\boldsymbol{\beta}$ and $\boldsymbol{X}$ again means vector multiplication, but this time the second "+" sign represents the addition of a number to a distribution. This results in a distribution in which all the outputs have been adjusted by the same amount. This kind of re-use of familiar symbols ("+" in this case) is very common in mathematics but can lead to confusion. Consequently, I am pointing it out.)

This model takes as input the same vector of five variable values but returns a probability distribution. To use this probability distribution we need to give it a particular possible value for y and it will return the probability density that this is the true value.

These examples illustrate how to specify a regression model. It is also useful to be able to specify the method of fitting, though in practice there aren't many of them and giving the name of the rule or method used is usually enough.

Specifying "fitting" rules

As an example, the OLS criterion could be defined as follows. First, there is a training set, ts, made from a number of cases. Since some cases could potentially be identical the training set is specified as having the type of a "bag" like this:

$ts : (X \times Y) \to \mathbb{N}$

The idea is that this gives the number of times each unique case occurs. The specification also involves a model family, mf, having a vector of parameters of type $\boldsymbol{Q}$. Supply values for the parameters and the model family returns a particular model predicting Y from X.

$mf : Q \to (X \to Y)$

The OLS criterion is only applicable to models making these best guess predictions.

The fitting criterion is to find a particular vector of parameter values that minimises the sum of the squares of the differences between the predictions and the actual values for the data in the training set.

$q : Q$

$$\neg\exists\, q' : Q \mid sum[(x,y) : X \times Y \mid (x,y) \in dom[ts] \cdot ts[(x,y)] \times (y - mf[q'][x])^2] \\ < sum[(x,y) : X \times Y \mid (x,y) \in dom[ts] \cdot ts[(x,y)] \times (y - mf[q][x])^2]$$

In this rule q is a particular value for the parameter vector such that there is no other parameter vector value that gives a lower sum of squared differences. There could be more than one such vector.

The MLE criterion can be specified in a similar way. The training set is the same but the models provide a distribution rather than just a best guess.

$ts : (X \times Y) \to \mathbb{N}$

$mf : Q \to (X \to (Y \to \mathbb{R}))$

$q : Q$

$$\neg\exists\, q' : Q \mid prod[(x,y) : X \times Y \mid (x,y) \in dom[ts] \cdot mf[q'][x][y]^{ts[(x,y)]}] \\ > prod[(x,y) : X \times Y \mid (x,y) \in dom[ts] \cdot mf[q][x][y]^{ts[(x,y)]}]$$

Again, there could be more than one best vector for the parameters. Implicit in this specification is the assumption that the cases in the training set are independent of each other.

The modern Bayesian approach to combining data with models is very different because, instead of trying to pick one best value for the parameter vector, the relative probabilities of all possible parameter vector values are represented and modified through consideration of the data. The training set and model family are the same as for the MLE criterion:

$ts : (X \times Y) \to \mathbb{N}$

$mf : Q \to (X \to (Y \to \mathbb{R}))$

However, the representation of parameters is very different. There is a set of possible values for the parameters and then two probability (or probability density) distributions, one to represent the position before considering the training set and the other the position afterwards.

$qs : \mathbb{P}\, Q$

$prior : Q \to \mathbb{R}$

$post : Q \to \mathbb{R}$

$dom[prior] = qs$

$dom[post] = qs$

The rule that determines the final distribution over the possible values for Q is:

$$\forall\, q : Q \mid q \in qs \cdot post[q]$$
$$= prior[q] \times \frac{prod[(x,y): X \times Y \mid (x,y) \in dom[ts] \cdot mf[q][x][y]^{ts[(x,y)]}]}{sum[q': Q \mid q' \in qs \cdot prod[(x,y): X \times Y \mid (x,y) \in dom[ts] \cdot mf[q'][x][y]^{ts[(x,y)]}]]}$$

Justifying "fitting" rules

The mathematics used to justify particular models, "fitting" criteria, and "fitting" algorithms is where assumptions about the data populations for which models are developed are relevant.

References

Spivey, J.M. (1989). The Z Notation. Prentice-Hall, Englewood Cliffs, NJ. Available online at: http://spivey.oriel.ox.ac.uk/mike/zrm/zrm.pdf

Appendix

In defining the criteria for model "fitting" I represented the training set as a bag of X and Y values. A slightly more familiar looking approach is to represent the training set with a sequence of X values and a sequence of Y values. This way the familiar index i is involved.

Here is the OLS criterion done in this way, with x and y taking the place of ts:

$x : seq\, X$

$y : seq\, Y$

$mf : Q \to (X \to Y)$

$q : Q$

$$\neg \exists\, q': Q \mid sum[i : \mathbb{N} \mid i \in 1..\#x \cdot \left(y[i] - mf[q'][x[i]]\right)^2] < sum[i : \mathbb{N} \mid i \in 1..\#x \cdot (t[i] - mf[q][x[i]])^2]$$

Another conventional refinement is to use the $argmin$ function, which returns the set of arguments giving the minimum value for the function. In this case that would be something like this:

$q \in argmin[q' : Q \cdot sum[i : \mathbb{N} \mid i \in 1..\#x \cdot (t[i] - mf[q'][x[i]])^2]]$

And here is the MLE criterion:

$x : seq\ X$

$y : seq\ Y$

$mf : Q \to (X \to (Y \to \mathbb{R}))$

$q : Q$

$\neg \exists\ q' : Q \mid prod[i : \mathbb{N} \mid i \in 1..\#x \cdot mf[q'][x[i]][y[i]] > prod[i : \mathbb{N} \mid i \in 1..\#x \cdot mf[q][x][y]]$

Or using $argmax$,

$q \in argmax[q' : Q \cdot prod[i : \mathbb{N} \mid i \in 1..\#x \cdot mf[q'][x[i]][y[i]]]$

And, finally, the Bayesian approach:

$x : seq\ X$

$y : seq\ Y$

$mf : Q \to (X \to (Y \to \mathbb{R}))$

$qs : \mathbb{P}\ Q$

$prior : Q \to \mathbb{R}$

$post : Q \to \mathbb{R}$

$dom[prior] = qs$

$dom[post] = qs$

$$\forall\ q : Q \mid q \in qs \cdot post[q] = prior[q] \times \frac{prod[i : \mathbb{N} \mid i \in 1..\#x \cdot mf[q][x[i]][y[i]]]}{sum[q' : Q \mid q' \in qs \cdot prod[i : \mathbb{N} \mid i \in 1..\#x \cdot mf[q'][x][y]]]}$$