

Validating Learning Algorithms

Noel Welsh

18 October 2010

1 Overfitting and Generalisation

Suppose we are considering several models for a learning problem. How should we select between them? So far, when choosing a hypothesis from a model class we have chosen the hypothesis that maximises the likelihood. We might extend this to choosing between models, say by choosing the model for that has the highest likelihood hypothesis. It turns out that this is not, in general, a good thing to do, due to the problem of overfitting.

For concreteness imagine we are considering polynomial regression models of the type $h_{\beta}(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \dots + \beta_k x^k$. As we increase k the model can get a closer and closer fit to any data we have. At some point the model will start to fit the noise in the data, rather than the true trend. This known as *overfitting*. This is illustrated in 1. Polynomials of degree 1, 2, 5, and 20 are fit to twenty data points generated from the natural logarithm and corrupted with normal noise. The quadratic (degree 2) polynomial comes closest to fitting the true trend. The polynomial of degree 20 exactly fits all the data but bears little resemblance to the underlying process generating the data.

Overfitting would not be a problem if we only ever had the data we used to train the model. In general, however, we are interested in the performance of the model on data that we have not yet seen. This is known as *generalisation*. A model that overfits will generalise poorly. Imagine if we could access all the data our model would ever encounter. If this was the case we could definitively calculate its performance. In our case the measure we use is likelihood, though other measures are in common use. We call the performance over all possible data the *generalisation error*.

2 Validating Learning Algorithms

The example in 1 is simple enough that we can tell the best model from the graph. In more complicated cases it would be nice to have an automated method that gives some idea of the generalisation performance of the models under consideration. The process of assessing the generalisation performance is called *validation*.

For many models we can derive bounds on the generalisation error. In this section, however, we will look at a computational method for estimating generalisation error. This is attractive as we can apply the validation algorithm to models for which we

cannot derive a bound, and it may give a better estimate of the generalisation error than any bound we have available.

2.1 Hold-out Cross Validation

The simplest method is to partition the available data so that, say, 70% is used for training and the remaining 30% is used to assess the models after training is complete. The data that is set aside is known as a *hold-out set*. The error on the hold-out set is used as our estimator of generalisation performance. If we can choose only one model we would choose the one with maximum likelihood on the hold-out set. Normally we would then train that model on the complete data set (training and hold-out set).

2.2 k -fold Cross Validation

Hold-out cross validation assumes data is plentiful, so we can afford to not use a fairly large proportion of data for training. This is not always the case, as for example in mobile robotics experiments that are time consuming to repeat. A method that makes better use of data is *k-fold cross validation*.

In k -fold cross validation we split our data set into k equal size disjoint sets. We then hold out one of these sets and perform hold-out cross validation. Then we put the hold-out set back into the training data and choose a different hold-out set, repeating hold-out cross validation. This process repeats till we have used each of our k sets as a hold-out set.

From k -fold cross validation we can estimate the generalisation error by averaging the generalisation error we found in each iteration of hold-out cross validation. As in hold-out cross validation once we have found the best model we typically train it on all the data.

A typical value for k is 5 or 10. In the limit, where each hold-out set consists of a single example from our data set, we call the algorithm *leave-one-out cross validation*. As k increases so does the amount of computation we must expend.

2.3 Assessing a Single Model

So far we have discussed validation in the context of choosing between models. It is equally valid to use it to assess a single model.

3 Notes

Statistical learning theory (SLT) is the name of the sub-field within machine learning that is concerned with developing bounds on generalisation (amongst other things). If you're interested in learning more about SLT a good place to start is Robert Nowak's course notes or course notes from Shai Ben-David and Shai Shalev-Shwartz.

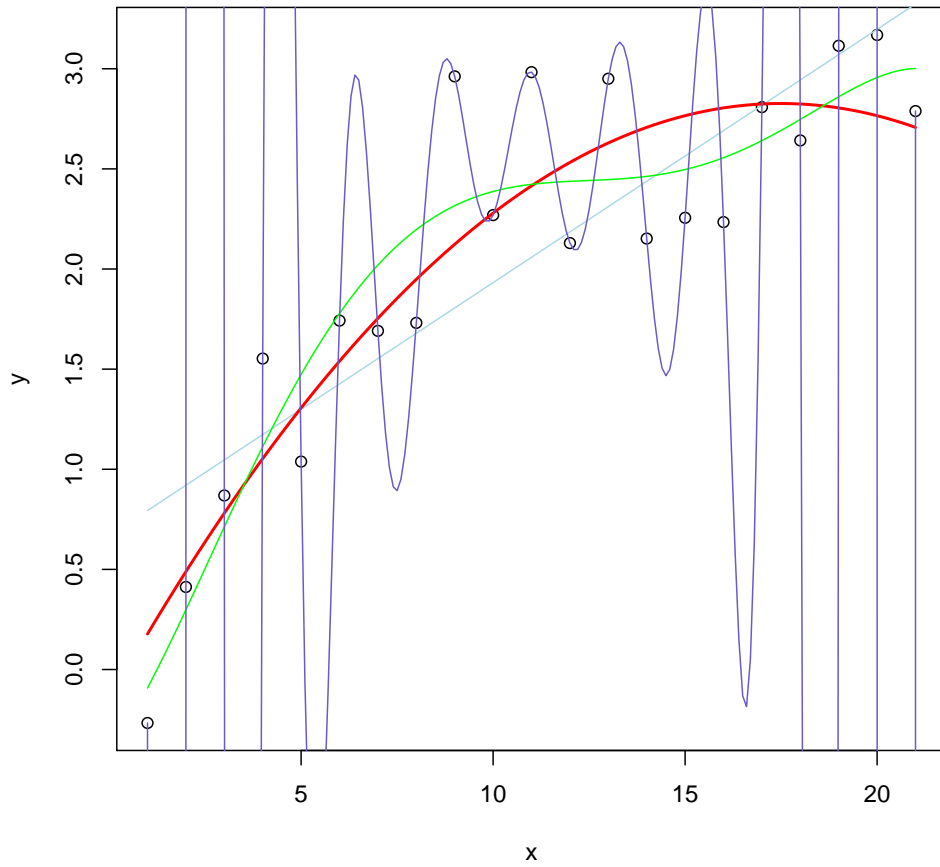


Figure 1: Example of fitting twenty points with a first, second, fifth, and twenty degree polynomial. The second degree polynomial best balances fit and model complexity.