Statistical Ideas for Selecting Network Architectures

Brian D. Ripley

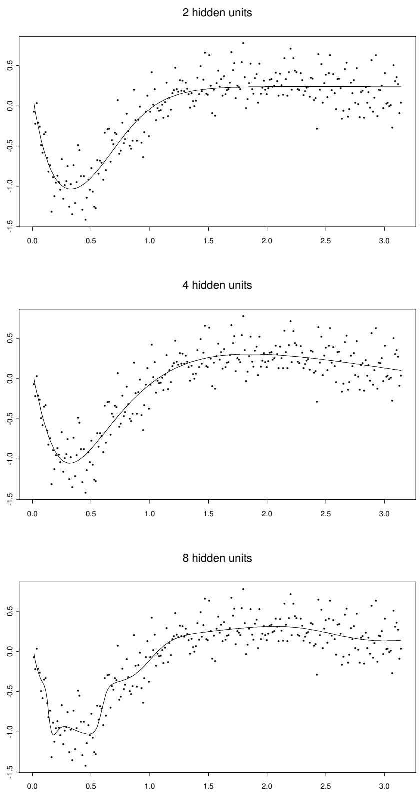
Professor of Applied Statistics
University of Oxford

Programme

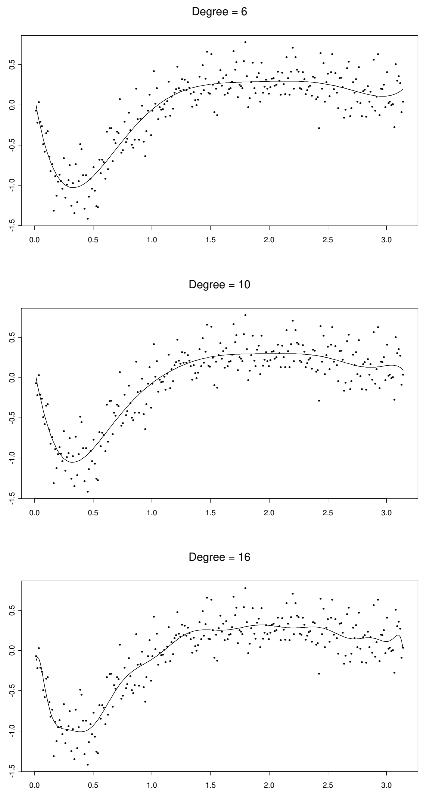
- Special considerations for neural networks
- Cross-validation
- Regularization (e.g. weight decay)
- AIC, NIC, Moody's p_{eff}, …
- BIC and Bayes factors
- Combining models

For simplicity, all the examples are feed-forward neural networks (MLPs), but the principles apply much more widely, including to RBFs.

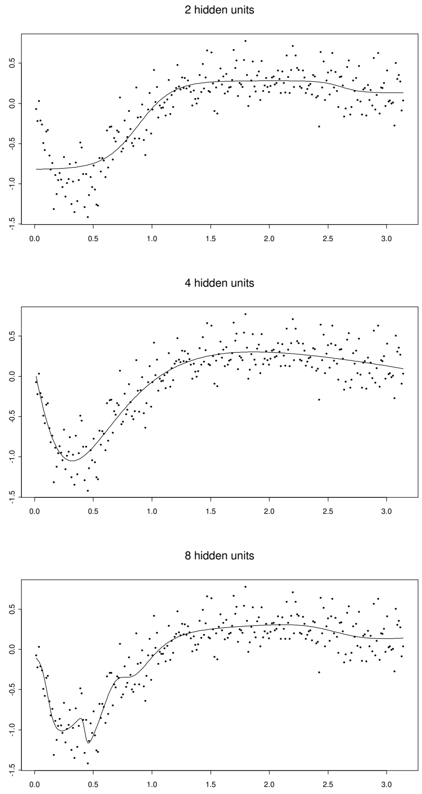
What's different about neural networks, compared to curve-fitting as done in statistics?



Fits with feed-forward neural networks.



Fits with polynomials.



More fits with feed-forward neural networks.

Differences

Main differences are:

- Multiple local minima are common (especially for large models).
- Over-fitting is controlled by non-likelihood fitting.
- Sequence of possible architectures is not necessarily nested.

Consequence:

Everyone fitting polynomials will get the same answer, but almost no two people fitting neural networks will. (*Exception:* networks without hidden layers.)

Many statistical ideas (including those used in MARS and projection pursuit regression) are based on nested maximum-likelihood fitting.

Selection Techniques

Many classical techniques work with *nested* families of models, e.g. degree of a polynomial regression, number of hidden units in a 'raw' FFNN.

- *Forwards selection* Add terms until the fit is good enough (cascade correlation).
- *Backwards selection* Build a large model then drop terms (projection pursuit regression, MARS, …).
- *Stepwise selection* Try both adding and dropping at each step.

Larger models will fit better, so need either significance tests or penalty for complexity. Many such

(a) *AIC* Akaike's 'An Information Criterion'

(b) *BIC* 'Bayesian Information Criterion'

(c) *MDL/MML* 'Minimum Description (or Message) Length— Rissanen and Wallace schools.

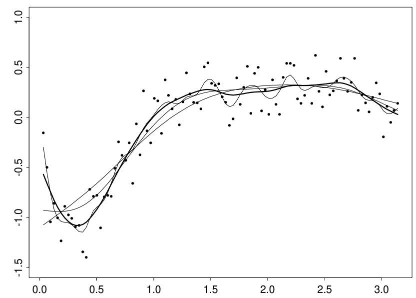
These work for non-nested models, but a search in model space is needed.

Smoothing Splines

Numerical analysts know that very flexible families of functions will overfit, so they add a penalty, for example

$$\sum_{i=1}^{n} [y_i - f(x_i)]^2 + \lambda \int f''(u)^2 \, du$$

This is minimized over *all* functions f by a (cubic) smoothing spline. It remains to choose λ, which controls the degree of smoothness.



Smoothing splines with various degrees of regularization. The bold line was chosen as the best fit.

Cross-Validation

Suppose we have:

- A training set $\mathcal{T}$ used to fit weights.
- A validation set used to select architectures (models).
- A test set used to estimate performance of selected model.

We can then use the performance on the *validation* set to choose the best model. ('Hold-out' method of Highleyman, 1962.)

Wasteful of data, so split the training set into V parts. Use $(V - 1)$ of these to train, and test the performance on the Vth part. Do this for all V parts. This creates a pseudo test set of size n. The performance measure is (almost) unbiased, since the parts are (almost) independent, but it is for performance at size $(1 - 1/V)n$.

Idea from the mid-1970s (M. Stone, Geisser, …).

How do we choose V? Traditionally *leave-one-out*, so $V = n$. This gives very variable estimates of performance, and can be seriously wrong. Often better to choose $V \approx \sqrt{n}$ or 10, and much cheaper computationally.

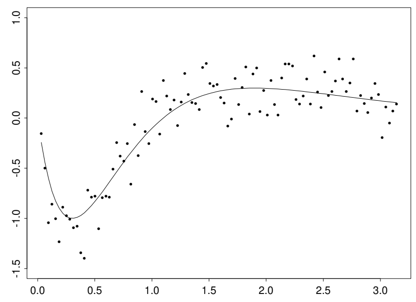
This estimates the effect of over-fitting by $O(1/\sqrt{(n)})$ perturbations of the problem, the order of the effects of variation between training sets. The *Bootstrap* is based on same notion.

Caveat

(Cross-)validation applies to a well-defined procedure, so troubles with multiple local minima, early stopping and so on.

Demonstration example

Taken from Wahba & Wold (1975) where it illustrates smoothing splines.

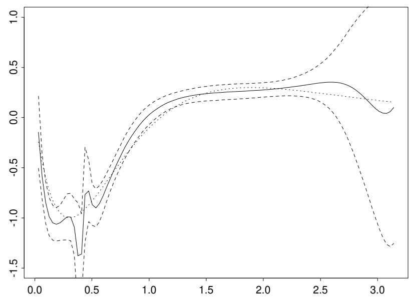


100 data points from a curve-fitting problem, with the true curve.

Throughout we assume that the true noise variance, $(0.2)^2$, is known.

Regression Neural Networks

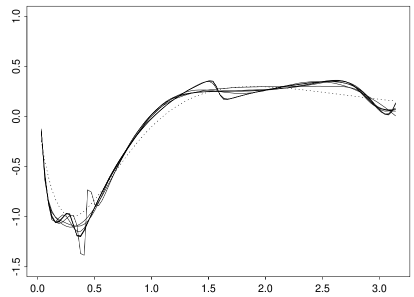
Model selection amounts to choosing the number of hidden units.



Curve fitted by 1-8-1 neural network, with ± 2 standard error bands and true curve (dotted).

The error limits come form standard regression theory, and are based on local linearization.

One problem is multiple local minima:



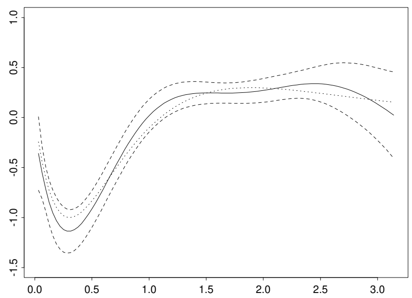
1-8-1 neural networks with different starting weights.

We clearly have over-fitting. Many ways to avoid this. Perhaps the best is to use *regularization* as used for smoothing splines. Somewhat easier for neural networks to use *weight decay*:

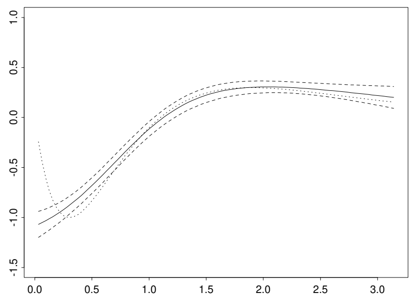
$$\sum_{i=1}^{n} [y_i - f(x_i; \mathbf{w})]^2 + \lambda \sum_{ij} w_{ij}^2$$

with the inputs and outputs suitably scaled.

Effect of Weight Decay

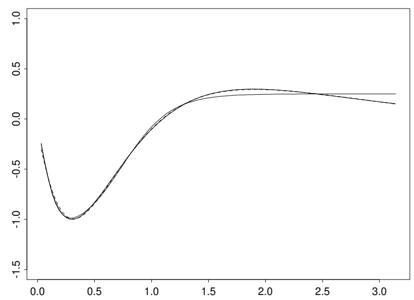


Curve fitted by 1-8-1 neural network, with ± 2 standard error bands and true curve $\lambda = 10^{-3}$ (above) and $\lambda = 0.1$ (below).



Bias/variance Compromises

We now have two ways to control the fit, the number of hidden units and λ, the degree of regularization. The true curve is not exactly representable by our model, so we have some bias.



Fitting to the true curve with 2 and 3 hidden units, and 8 units with $\lambda = 10^{-3}$.

Choosing fewer hidden units leads to more bias, as does adding a regularizer. But it also reduces their variability, so reduces the mean-square error. Can manipulate λ and the number of hidden units as a function of n to get the right compromise for very large n. Doesn't help in real problems!

AIC, NIC, p_{eff}

Akaike's (1973, 4) 'An Information Criterion'.
ML measure of fit is the deviance

$$2 \sum \log p(X_i, Y_i) / p(X_i, Y_i; \widehat{\theta})$$

on the training set (the supplied data). Consider a test set, that is an independent sample from the same process of the same size. We would expect the deviance for the test set to be larger.

AIC corrects this bias:

$$E\left[\text{deviance}_{\text{test}} - \text{deviance}_{\text{train}}\right] \sim 2p$$

for p parameters and for large n *under the model*. Choose model(s) which have small values of

$$\text{deviance}_{\text{train}} + 2p$$

Since AIC is based on maximum likelihood estimations, it has nothing to say in our example if regularization is used.

If we drop the requirements that the model is true, and that fitting is by ML, we find

$$E\left[\text{deviance}_{\text{test}} - \text{deviance}_{\text{train}}\right] \sim 2p_{\text{eff}}$$

where $\qquad p_{\text{eff}} = \text{tr}[KJ^{-1}] \qquad$ and

$$J = -E\frac{\partial^2 \log p(X_i, Y_i; \theta_0)}{\partial\theta\, \partial\theta^T}, K = \text{Var}\frac{\partial \log p(X_i, Y_i; \theta_0)}{\partial\theta}$$

and θ_0 is the 'least false' value of θ, that is minimizes

$$-E \log p(X, Y; \theta)$$

This is due to Moody (1991, 1992) and Murata, Yoshizawa & Amari (1991,3,4) (hence NIC). Derived (but not stated) by Stone (1977). Proof is easy, based on second-order Taylor expansions of the log-likelihood, *about a single local maximum.*

Note that AIC, NIC only correct the bias as an estimate of the expected test-set deviance. We have to consider the fluctuations. For large n, AIC-like procedures never choose too small a model, but may choose too large a one.

Asymptotically LOO CV is the same as NIC (Stone, 1977), and often accurately so in small samples too. NIC is much less work!

BIC and Bayes Factors

'BIC' often refers to the penalty $p \log n$. An approximation to the use of *Bayes factors*.

The predictive Bayesian approach assesses models via $P(\text{model} \mid \mathcal{T})$, integrating out 'weights' θ.. Now

$$P(\text{model} \mid \mathcal{T}) \propto P(\text{model}) P(\mathcal{T} \mid \text{model})$$

Strictly we average predictive distributions with weights $P(\text{model} \mid \mathcal{T})$. But if we choose, a crucial factor is

$$B_{st} = \frac{P(\mathcal{T} \mid \text{model}_s)}{P(\mathcal{T} \mid \text{model}_t)}$$

called the Bayes factor, since

$$\frac{P(\text{model}_s \mid \mathcal{T})}{P(\text{model}_t \mid \mathcal{T})} = B_{st} \frac{P(\text{model}_s)}{P(\text{model}_t)}$$

Often

$$\begin{aligned} \log P(\mathcal{T} \mid \text{model}) \;\approx\; & \log \ell(\widehat{\theta}; \mathcal{T}) + \log p(\widehat{\theta}) \\ & + \tfrac{p}{2} \log 2\pi - \tfrac{1}{2} \log |H| \end{aligned}$$

for H the Hessian of the sum of the first two terms, so can be approximated.

The last approximation depends on a single local maximum, but we can sum up over multiple local maxima.

Approximations to this approach have been applied by David MacKay. It has a long history in statistics, e.g. the book by Aitchison & Dunsmore (1975).

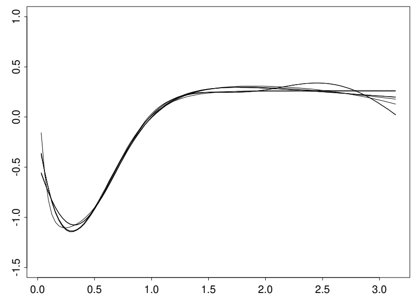
In example we look at $\sigma^2 \times \log P(\mathcal{T} \mid \text{model})$, since $\sigma^2 \times \log \ell(\widehat{\theta}; \mathcal{T})$ is the residual sum of squares.

Results for Example

Size	λ	SSq	SSq$_{CV}$	p_{eff}	NIC	Bayes
2	0	3.47	4.29	7	4.03	—
2	10^{-3}	3.57	4.78	5	3.98	3.30
3	0	4.12	4.81	10	4.92	—
3	10^{-3}	3.40	4.54	6	3.93	3.30
3	10^{-2}	3.61	4.91	6	4.11	3.72
3	10^{-1}	5.40	6.20	6	5.94	6.08
5	0	3.14	4.57	16	4.42	—
5	10^{-3}	3.21	4.51	17	4.59	3.38
5	10^{-2}	3.61	4.91	6	4.11	4.11
5	10^{-1}	5.38	6.18	6	5.92	5.98
8	0	2.89	5.35	25	4.89	—
8	10^{-3}	3.21	4.54	18	4.67	4.74
8	10^{-2}	3.58	4.94	7	4.16	4.63
8	10^{-1}	5.36	6.14	7	5.90	5.83

Column SSq$_{CV}$ is based on 10-fold cross-validation, the same sets for all the models.

Model Uncertainty



The fitted curves for all the well-supported models.

Bayesian approach averages predictions (densities) over models, so incorporates model uncertainty.

Finding the Bayes factors and weighted average is very time-consuming, *and* assumes the models are true. Stone (1974) suggested using LOO cross-validation to choose the weights in a weighted average, as have many people since.

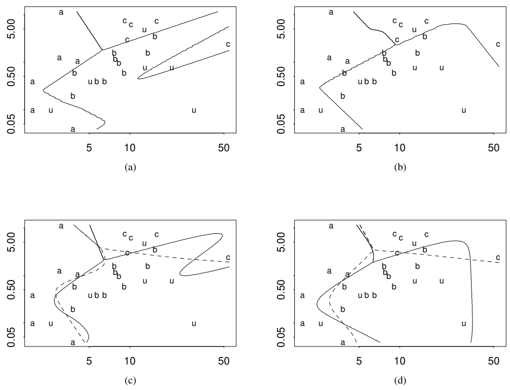
Clearly we should take a weighted average over local minima, and it *may* suffice to take an unweighted average.

Cushing's Syndrome

Data on diagnostic tests on patients with Cushing's syndrome, a hypertensive disorder associated with over-secretion of cortisol by the adrenal gland.

This dataset has three recognised types of the syndrome represented as a, b, c. (These encode 'adenoma', 'bilateral hyperplasia' and 'carcinoma'.)

The observations are urinary excretion rates (mg/24h) of the tetrahydrocortisone and pregnanetriol, and are considered on log scale. Those marked u are future examples to be classified.

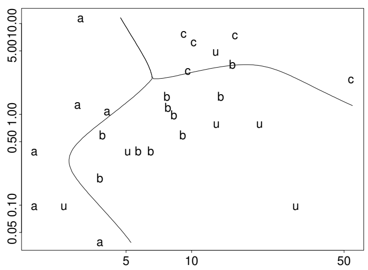


Neural network fits with two hidden units and connections from the input to output layer. Figures (a) and (b) are two solutions without weight decay which fit perfectly. Figures (c) and (d) show two local minima each, for $\lambda = 0.001$ and $\lambda = 0.01$ respectively. The dashed lines correspond to the local minimum that fits less well.

We can compare the seventeen local minima found with $\lambda = 0.01$. Six of them carry 5% or more of the total mass, with fitted values and weights

value	4.544	5.928	6.198	4.502	4.269	4.180
%	24.4	21.1	17.5	6.9	6.1	5.7

The second and sixth are shown at (d).



Predictive neural network fit to the data on Cushing's syndrome with two hidden units and $\lambda = 0.01$.

References

Ripley, B. D. (1995) Statistical ideas for selecting network architectures. In *Neural Networks: Artificial Intelligence and Industrial Applications*, eds B. Kappen & S. Gielen, pp. 183–190. Springer.
[neural/papers/SNN.ps.Z]

Ripley, B. D. (1996) *Pattern Recognition and Neural Networks*. Cambridge University Press. ISBN 0 521 48086 7.
[http://www.stats.ox.ac.uk/~ripley/PRbook]

A PostScript version of the slides of the talk is at

`bdr/NIPS/selecting.ps.gz`.

On-line versions obtainable by ftp from

`markov.stats.ox.ac.uk:/pub/`

or via WWW at URL

`http://www.stats.ox.ac.uk/pub/`.