A Pragmatic Bayesian Approach to Predictive Uncertainty

Iain Murray and Edward Snelson

Gatsby Computational Neuroscience Unit, University College London,
London WC1N 3AR, UK
{i.murray, snelson}@gatsby.ucl.ac.uk
http://www.gatsby.ucl.ac.uk/

Abstract. We describe an approach to regression based on building a probabilistic model with the aid of visualization. The "stereopsis" data set in the predictive uncertainty challenge is used as a case study, for which we constructed a mixture of neural network experts model. We describe both the ideal Bayesian approach and computational shortcuts required to obtain timely results.

1 Introduction

We describe our treatment of the "stereopsis" regression data set in the predictive uncertainty challenge. Our aim was to construct an appropriate statistical model of the data, and use Bayesian inference to form the required predictive distributions given the data. Our starting point was some simple exploratory visualization of the data.

The stereopsis data set is very amenable to visualization, as it has only four input dimensions, and structure quickly reveals itself. For example plotting various input dimensions against each other, as in Figure 1(a), shows some clear clustering in the input space. There are 10 distinct branches of points, each with some interesting substructure. Figure 1(b) shows a one dimensional projection of the training inputs plotted against the training outputs. This projection was made by doing a least squares linear fit to the training data, $\hat{\mathbf{w}} = \mathrm{argmin}_{\mathbf{w}} \sum_n (y^{(n)} - \mathbf{w}^\top \mathbf{x}^{(n)})^2$, and plotting the targets y against projected inputs $\hat{\mathbf{w}}^\top \mathbf{x}$. The training outputs also seem to be grouped into 10 discrete clusters, so one might guess that there is a correspondence between the clustering in input space, and the clustering in output space. Further visualization confirmed that this was the case. Zooming in on a cluster in Figure 1(b), it becomes clear that there is also substructure relevant for regression within each output cluster.

When we see obvious structure from simple visualizations like Figure 1, it makes us curious about the data generating process. Are the input points sampled from some natural distribution on the space of possible inputs? In which case we might not worry about generalizing well to input points far away from those already observed. Alternatively the clustering in input and output space may be an artifact of a particular choice of experiments that have been done. Then some new test positions may lie in other parts of the input space outside

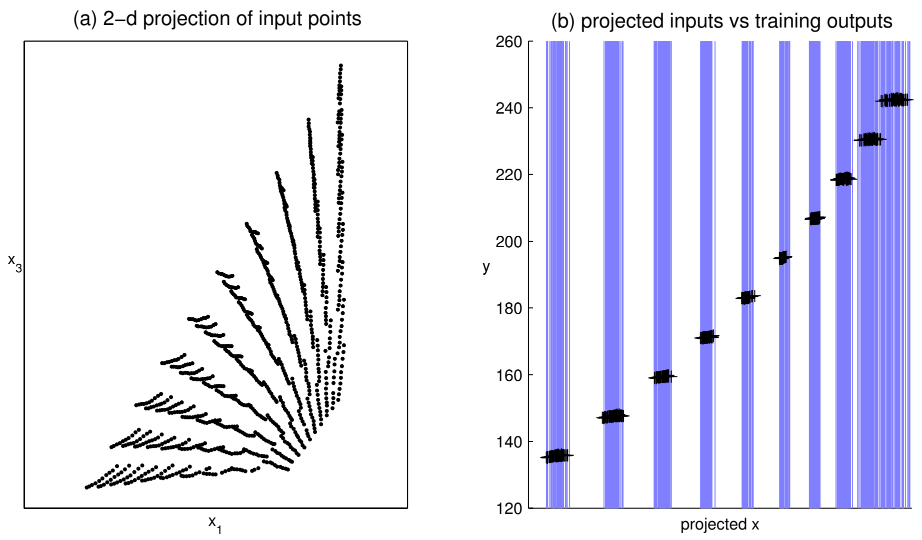

Fig. 1. Visualization of stereopsis dataset. (a) The first and third dimensions of the input space are plotted against each other. (b) The training outputs are plotted against $\hat{\mathbf{w}}^\top \mathbf{x}$, the input points projected onto the least squares linear regression weight vector. The test input projections onto the same weight vector are plotted as the vertical lines.

the regions observed in the training data. In which case, we would need to concentrate on building a global model that extrapolates well outside the observed clusters with appropriate uncertainties. In a real world task we would probably have information regarding the locations of future predictions.

Unknown to us, the 'stereopsis' task was to infer the depth of an object from stereo image information [1]. The data were generated by attaching an LED to a robot arm and recording the spatial location via the arm's spatial encoders and the LED image positions on two stereo cameras. The robot arm was then moved in a structured way in different discrete planes of depth whilst data were recorded. With this small amount of information it is clear where the almost discrete nature of the outputs (the depths) comes from, and also the clustering in the inputs. It is also clear that for a model to be useful it must be able to extrapolate away from the training data clusters, as we will need to be able to predict the whole range of depths well.

The nature of a machine learning competition is a little different. The data are presented with no information except a slight clue in the title! Clearly with all the structure visible, we could still make a good guess that this was caused by a certain choice of experimental sampling. We are also given the test inputs; a projection of these is shown in Figure 1(b). From this it seems plausible, as in most machine learning competitions, that the test data have been produced from the same sampling distribution as the training data. Therefore, in order to perform well in the competition it made sense to model the cluster structure. This is described in the next section.

2 Mixture of Experts Model

As described above, we based our model on intuitions gained from visualizing the training data and the test input locations. Further visualization of each of the ten clusters in figure 1 showed that the targets seemed to vary smoothly with any linear projection of the input space. We chose to model the targets within each cluster as noisy observations around a smooth function of the input space. To complete the model we defined a conditional distribution for belonging to a particular cluster given its input location. This is a mixture of experts model [2], which we refer to as "$\mathcal{H}$":

$$p(y|x, \theta, \mathcal{H}) \equiv \sum_k p(y|x, \mathbf{w}_k, \mathcal{H}_k) p(k|x, \mathbf{w}_{\text{gate}}, \mathcal{H}_{\text{gate}}), \tag{1}$$

where k indexes the experts and $\theta \equiv \{\mathbf{w}_k, \mathbf{w}_{\text{gate}}\}$ summarizes any free parameters in the models. We wanted flexible models for both the gating model, $\mathcal{H}_{\text{gate}}$, specifying the conditional probability of choosing an expert given input location, and for each expert's regression model $\mathcal{H}_k$. The groups over which the gating model puts a distribution have a clear ordering, see figure 1(b). This makes learning $p(k|x)$ an ordinal regression task, although for simplicity we considered it to be a standard multi-way classification problem. We chose to use neural networks [3] for both the regression and classification problems. Each neural network had a single layer of 15 units; we thought this would be sufficiently flexible while being manageable in the time available. Without further knowledge about the data, any flexible models such as Gaussian processes would have been equally sensible.

Each cluster in the training data only contains a small number of data points that will be useful for training that cluster's regression model $\mathcal{H}_k$. Therefore, there is a danger of over-fitting if we optimized the many parameters of the flexible models we chose. We could have chosen simpler models, but it might have been difficult to capture the non-linear structure we observed within the clusters. It is also possible that careful regularization could avoid over-fitting. Instead we decided to take our model seriously, and as far as possible perform the correct coherent inference given our assumptions; this is achieved by the ideal Bayesian approach.

3 Ideal Bayesian Inference

Our target is $p(y|x, D, \mathcal{H})$, the predictive distribution for an output y sampled at a new location x given previously observed data D and our modeling assumptions, $\mathcal{H}$. Our model has many free parameters, θ, corresponding to the weights in the classifier $\mathcal{H}_{\text{gate}}$ and each of the regression neural networks $\mathcal{H}_k$. As θ are unknown, we must marginalize these out:

$$p(y|x, D, \mathcal{H}) = \int p(y, \theta|x, D, \mathcal{H}) \, \mathrm{d}\theta = \int p(y|x, \theta, \mathcal{H}) p(\theta|D, \mathcal{H}) \, \mathrm{d}\theta. \tag{2}$$

The integrand consists of two parts. The first, $p(y|x, \theta, \mathcal{H})$ from equation (1), describes the predictive distribution given known parameters. The second part is the posterior over parameters from Bayes' rule:

$$p(\theta|D, \mathcal{H}) = \frac{p(D|\theta, \mathcal{H})p(\theta|\mathcal{H})}{p(D|\mathcal{H})} \propto p(D|\theta, \mathcal{H})p(\theta|\mathcal{H}), \tag{3}$$

which requires a prior over parameters $p(\theta|\mathcal{H})$ and a likelihood, which under our i.i.d. model is a product of terms:

$$p(D|\theta, \mathcal{H}) = \prod_n p(y^{(n)}|x^{(n)}, \theta, \mathcal{H}). \tag{4}$$

Each term, specified in (1), involves summing over the latent class assignment $k^{(n)}$. In order to obtain results more quickly we chose to approximate (1) by assuming we knew the class assignments for the training data based on the clusters in figure 1(b). While this is not the ideal procedure, it saved time and seemed a reasonable approximation given how well separated the targets were.

For the prior over parameters $p(\theta|\mathcal{H})$ we used the same hierarchical prior as in the neural network regression and classification examples in Neal's FBM documentation [4]. It is possible that the regression experts $p(y|x, \mathbf{w}_k, \mathcal{H}_k)$ should be related, especially now we know how the data were generated. Therefore it would make sense to introduce a priori correlations between the parameters, $\mathbf{w}_k$, of the experts. We chose not to do this; we assume our experts obtained parameters independently. As a result we probably did not make best use of the data.

The above theory says that the predictive distribution (2) is available without reference to how the input locations x were chosen, or what the predictions will be used for. Normally a loss function would only be necessary if we wanted to use the predictive distribution, eg for making a decision. Then, given the loss function $\mathcal{L}(y_{\text{guess}}, y_{\text{true}})$, which specifies the penalty for predicting y_{guess} when the test target is y_{true}, we would minimize our expected loss:

$$y_{\text{guess}} = \text{argmin} \int \mathcal{L}(y_{\text{guess}}, y)\ p(y|x, D, \mathcal{H})\ \mathrm{d}y, \tag{5}$$

where again, the predictive distribution is independent of the loss function. In the challenge the loss function depended on the distribution itself and we had to decide which quantiles to report. It turns out that both loss functions used in this challenge, mean squared error (MSE) and negative log predictive density (NLPD) have a consistency property: the expected loss is minimized by reporting the predictive distribution that actually reflects our beliefs.

In practice we will experience computational difficulties. As is often the case, the posterior in (3) has no simple form and the integral in (2) is intractable. A variety of approximate approaches exist. When using an approximation, the predictive distribution that results is, frankly, wrong: it does not correspond to the correct rational inference for combining our model with the data. The seriousness of this problem may depend on the loss function, although we did not use this to guide our approximation.

4 Monte Carlo Approximation

Here we present a standard Monte Carlo approach for predictive distributions. The first step is to draw S samples from the posterior distribution over parameters (3). These could be used to approximate the integral in (2):

$$\begin{aligned} p(y|x, D, \mathcal{H}) &= \int p(y|x, \theta, \mathcal{H}) p(\theta|D, \mathcal{H}) \,\mathrm{d}\theta \\ &\approx \sum_{s=1}^{S} p(y|x, \theta^{(s)}, \mathcal{H}), \qquad \theta^{(s)} \sim p(\theta|D, \mathcal{H}). \end{aligned} \tag{6}$$

However, this does not easily give us the quantiles required for reporting the whole distribution. Draws from the predictive distribution, $y^{(s)} \sim p(y|x, D, \mathcal{H})$, will be more useful. These can be obtained by first drawing a set of parameters, $\theta^{(s)} \sim p(\theta|D, \mathcal{H})$, then drawing $y^{(s)} \sim p(y|x, \theta^{(s)}, \mathcal{H})$. Quantiles of the predictive distribution may be approximated by drawing many samples and using the empirical quantiles of the set of samples. For implementational reasons we found approximate quantiles for each expert's distribution $p(y|x, D, \mathcal{H}_k)$ separately, and then combined these distributions using the mixing fractions $p(k|x, D, \mathcal{H}_{\text{gate}})$.

All of these sampling procedures are frequentist not Bayesian in nature. They give unbiased procedures, which are correct in the limit of an infinite number of samples under fairly general conditions. If our procedures used independent samples, then the errors in estimators of expectations are usually nearly Gaussian distributed and estimators of quantiles are also well understood as Beta distributed “order statistics”.

In this case, independent sampling is intractable. Markov Chain Monte Carlo (MCMC) methods allow drawing correlated samples from $p(\theta|D, \mathcal{H})$ [5]. Also, given a valuable parameter sample, $\theta^{(s)}$, it makes sense to draw multiple samples from $p(y|x, \theta^{(s)}, \mathcal{H})$, which are correlated samples from $p(y|x, D, \mathcal{H})$. Diagnosing the errors from these approximations remains a difficult problem; our approach was pragmatic. We ran a trial run of MCMC using only the training set and checked our results made reasonable predictions on the validation set. We then performed a longer run using both the training and validation sets for our final results. Strictly the Bayesian framework does not need a separate validation set at any stage, but we wished to check that the approximate inference procedure gave sensible results.

All of the above approximations were performed on neural networks with hierarchical priors by Radford Neal's FBM software [4].

5 Discussion

We exploited an artificial cluster structure of the stereopsis data, which should not necessarily be done in a real world modeling situation, eg [1]. This suggests some alternative formats for future competitions. Firstly, the standard assumption that the test data should come from the same *input* distribution as the training data could be relaxed. Secondly some information about the data could be given to competitors to guide modeling. This goes somewhat against a common view that a machine learning algorithm should be a general purpose black box. Competitors would need to tailor their methods using the information given about the training and test data generating processes. From this format it may be more difficult to achieve a consensus as to which machine learning algorithm performs best generally, but it may be more realistic, and some useful modeling approaches may come out of it.

These reservations aside, the approach outlined in the previous sections performed well in the competition, although was the mixture of experts approach really necessary? Initially we had tried to construct predictive distributions using only a single, global neural network regression model (we also tried Gaussian processes). Our visualizations indicated that the resulting error bars were too conservative. The added flexibility in the mixture model gave better NLPD scores on the validation set.

The ideal Bayesian approach assumes that all dependence on the data is contained in the likelihood. By looking at the data before constructing our model we were effectively using the data twice and risked over-fitting. As a result it is likely that we would not perform well on points drawn from a different input distribution. Fortunately for us, the competition did not reflect what often happens when a system is deployed: the input distribution changes.

It is unlikely we would have considered the model we used without any visualization. In a real application we would still recommend looking at the data to suggest suitable models. However, choosing one model by hand, as we did, is potentially dangerous. Better would be to consider a range of possibilities as well, eg: other settings for k and some simpler models. Predictions from different models can be combined quantitatively by Bayesian model averaging:

$$p(y|x, D) = \sum_i p(y|x, D, \mathcal{H}_i) p(\mathcal{H}_i|D). \tag{7}$$

This could be more robust than our approach using one model, if the clustering assumptions of our model turned out to be inappropriate for the test set.

Our approach scored an NLPD score considerably higher than other entrants to the competition. We believe the (nearly) Bayesian approach we followed was largely responsible. Firstly when regressing the individual clusters we integrated over uncertainty in the neural net parameters using MCMC, which avoided over-fitting. Secondly we were able to integrate over uncertainty in our classification parameters. This resulted in predictive distributions with ten modes, one for each expert, $\mathcal{H}_k$. Putting probability mass in all of these locations makes us robust to classification errors. If we had chosen to use only one expert for predictions we would risk obtaining an arbitrarily bad NLPD score.

Figure 2 shows on a log scale a typical predictive distribution given a new test input, from our competition submission. One of the experts $\mathcal{H}_k$ is favored over the others; it contributes the sharp spike close to $y = 220$. Notice how the predictions from the experts far from the most probable spike give broader, less certain predictions. This is because the new test input is far from the training inputs for those particular experts. The constant density between the spikes

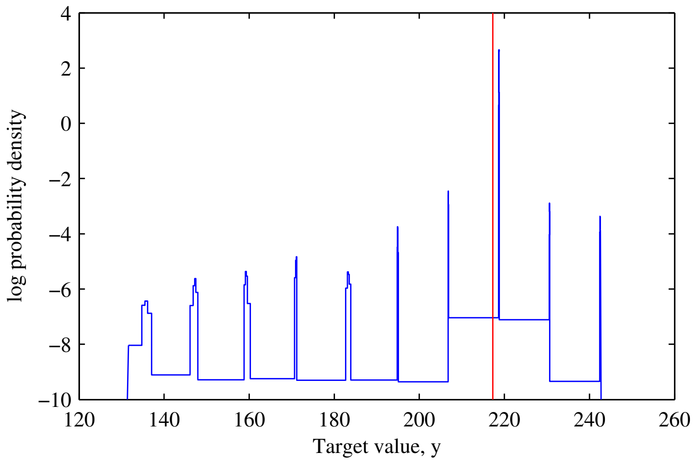

Fig. 2. A predictive log probability density constructed from quantiles submitted to the competition by us for a typical point in the test set. The vertical bar shows the corresponding mean of the predictive distribution.

results from an artifact of the way we combined the quantiles from each expert. The tail masses from adjacent experts was spread uniformly across the gap between them. We hoped that such artifacts would have low enough densities not to matter.

Also notice that the mean of the distribution in figure 2 does not coincide with any of the predictive spikes. This happens on most of our test set predictions, resulting in a poor mean squared error score; in fact we scored *last* on MSE score. The mean of a predictive distribution is fairly sensitive to small changes in the probability mass assigned to extreme predictions. In this case, small changes in the distribution of mass amongst the experts, $p(k|x, D, \mathcal{H}_{\text{gate}})$, can have a large effect. We made ourselves robust for NLPD score by placing mass on all modes, but at the expense of poor MSE score. The MSE score would have been most likely much improved if we had done hard classification.

6 Conclusions

This case study illustrated some of our opinions on how to construct models that capture predictive uncertainty:

– Visualization helps in understanding data when constructing probabilistic models, especially in the absence of any further domain knowledge.
– Bayesian inference provides a natural framework for finding predictive distributions given modeling assumptions. It can help avoid overfitting when limited data are available.
– Looking at the data before specifying modeling assumptions and using approximations both fall outside the Bayesian framework. By checking on a validation set, we found approximate Bayesian procedures still behaved robustly.

References

1. Sinz, F., Quiñonero-Candela, J., Bakir, G.H., Rasmussen, C.E., Franz, M.: Learning depth from stereo. In Rasmussen, C.E., Buelthoff, H.H., Giese, M.A., Schoelkopf, B., eds.: Proc. 26th DAGM Symposium, Springer (2004) 245–252
2. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the EM algorithm. Neural Computation **6** (1994) 181–214
3. Neal, R.M.: Bayesian Learning for Neural Networks. Number 118 in Lecture Notes in Statistics. Springer-Verlag, New York (1996)
4. Neal, R.M.: Flexible Bayesian modeling software (FBM). Available through `http://www.cs.toronto.edu/~radford/` (2003)
5. Neal, R.M.: Probabilistic inference using Markov chain Monte Carlo methods. Technical report, Dept. of Computer Science, University of Toronto (1993)