Bayesian model choice

Fully Bayesian procedures can't suffer from "overfitting" exactly, because parameters aren't *fitted*: Bayesian statistics only involves integrating or summing over uncertain parameters, not optimizing. The predictions can depend heavily on the model, and choice of prior however.

Previously we chose models — and parameters that controlled the complexity of models — by cross-validation. The Bayesian framework offers an alternative, using *marginal likelihoods*.

The problems we have instead of overfitting

In the Bayesian framework, we should use any knowledge we have. So if we knew (for some reason) that some points were observations of a 9th order polynomial, we would ideally use that model regardless of how much data we had. For example, given only 5 data points, we can't know where the underlying function is. However, predictions use a weighted average over all possible fits: our predictive distribution would be broad/uncertain, and centred around a sensible regularized interpolant.

That's not to say we have no problems when using Bayesian methods.

If the model is too simple, we saw in a previous lecture that the posterior distribution becomes sharply peaked around the least bad fit. As a result, we can be very confident about properties of a model, even if running some checks (such as looking at residuals) would show that the model is obviously in strong disagreement with the data.

We can also have problems when we use a model that's too complicated. As an extreme example for illustration, we could imagine fitting a function on $x \in [0, 1]$ with a million RBFs spaced evenly over that range with bandwidths $\sim 10^{-6}$. We can closely represent any reasonable function with this representation. However, given (say) 20 observations, most of the basis functions will be many bandwidths away from all of the observations. Thus, the posterior distribution over most of the coefficients will be similar to the prior (check: can you see why?). Except at locations that are nearly on top of the observed data, our predictions will be nearly the same as under the prior. We will learn slowly with this model.

Simple cases of probabilistic model choice

We have already used a simple form of probabilistic model comparison in Bayes classifiers. Given two fixed models $p(\mathbf{x} \,|\, y\!=\!1)$ and $p(\mathbf{x} \,|\, y\!=\!0)$, we could evaluate a feature vector under each and use Bayes' rule to express our beliefs, $P(y \,|\, \mathbf{x})$, about which model the features came from.

With Gaussian class models, there are different ways that a model can win a comparison. If a model has a tight distribution, then it will usually be the most probable model when observations are close to its mean, even if those observations could also have come from a broad distribution centred nearby. On the other hand, broad distributions become the most probable model for extreme feature vectors or outliers. Finally, while the prior probabilities of the class models have some effect, the likelihoods of the models often dominate.

My other favourite example of comparing broad and narrow models is dice with different numbers of sides. If I said that I chose a dice at random from a million-sided dice and a 10-sided dice and got a 5, you'd be pretty sure I'd rolled the 10-sided dice. That's partly because of priors: you don't think I could possibly own a million sided dice. But even if you thought that million-sided dice were just as common, you'd have the same view. For example, I could implement this game on a computer and show you the code:

```
    sides = [10, 1e6];
    dice = (0.5<rand()) + 1;
    outcome = floor(rand()*sides(dice)) + 1
```

Every time you see this code output a number between 1 and 10, you'll assume that it came from `dice=1`. While it's rare to get a small outcome like 5 with a million sided dice, it's no rarer than any other outcome, such as 63,823. However, the small outcomes are more easily generated under the alternative narrow model, so it wins the comparison.

Application to regression models

Why do we usually favour a simple fit over the model with a million narrow basis functions described above? It could be because of priors: we might favour simple models. But actually the world is complicated, and my prior beliefs are that many functions have many degrees of freedom. What would happen if we gave half of our prior mass to the model with a million narrow basis functions? It would still usually lose a model comparison.

A regression model has to distribute its prior mass over all of the possible regression surfaces that it can represent, or over all of its parameters. If a model can represent many different regression surfaces, only some of these will match the data. The probability that a model $\mathcal{M}$ assigns to some observations in a training set is:

$$p(\mathbf{y}\,|\,X,\mathcal{M}) = \int p(\mathbf{y},\mathbf{w}\,|\,X,\mathcal{M})\;\mathrm{d}\mathbf{w} = \int p(\mathbf{y}\,|\,X,\mathbf{w},\mathcal{M})\;p(\mathbf{w}\,|\,\mathcal{M})\;\mathrm{d}\mathbf{w},$$

where the parameters $\mathbf{w}$ are assumed unknown. Narrowly focussed models, where the mass of the prior distribution $p(\mathbf{w}\,|\,\mathcal{M})$ is concentrated on simple curves, will assign higher density to the outputs $\mathbf{y}$ observed in many natural datasets than the million narrow basis function model. The narrow basis function model can model smooth functions, but can also fit highly oscillating data — it's a broader model that can explain outliers, but that will usually lose to simpler models for well-behaved data.

The probability of the data under the model, given above, is the model's *marginal likelihood*, and can be used to score different models, instead of a cross-validation score. For Gaussian models, and some other models (usually with *conjugate priors*, discussed in Tutorial 6), we can compute the integral. Later in the course we also discuss how to approximate marginal likelihoods, where we can't solve the integral in closed form.

Application to hyper-parameters

The main challenge with model-selection is often setting real-valued parameters like the noise level, the typical size of the weights (their prior standard deviation), and the widths of some radial basis functions. These values are harder to cross-validate than simple discrete choices, and if we have too many of these parameters, we can't cross-validate them all.

Incidentally, in the million narrow RBFs example, the main problem wasn't that there were a million RBFs, it was that they were *narrow*. Linear regression will make reasonable predictions with many RBFs and only a few datapoints if the bandwidth parameter is broad and we regularize. So we usually don't worry about picking a precise number of basis functions, or hidden units in a neural network.[1]

The full Bayesian approach to prediction was described in the previous note. We integrate over all parameters we don't know. In a fully Bayesian approach, that integral would include noise levels, the standard deviation of the weights in the prior, and the widths of basis functions. Because the best setting of each of these values is unknown. However, computing integrals over all of these quantities can be difficult (we will return to methods to approximate such difficult integrals after studying Gaussian processes).

A simpler approach is to maximize some parameters, according to their marginal likelihood, the likelihood with most of the parameters integrated out. For example, in a linear regression model with prior

$$p(\mathbf{w}\,|\,\sigma_w) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \sigma_w^2\mathbb{I}),$$

and likelihood

$$p(y\,|\,\mathbf{x},\mathbf{w},\sigma_y) = \mathcal{N}(y; \mathbf{w}^\top\mathbf{x}, \sigma_y^2),$$

we can fit the *hyperparameters* σ_w and σ_y, parameters which specify the model, to maximize their marginal likelihood:

$$p(\mathbf{y}\,|\,X,\sigma_w,\sigma_y) = \int p(\mathbf{y},\mathbf{w}\,|\,X,\sigma_w,\sigma_y)\;\mathrm{d}\mathbf{w} = \int p(\mathbf{y}\,|\,X,\mathbf{w},\sigma_y)\;p(\mathbf{w}\,|\,\sigma_w)\;\mathrm{d}\mathbf{w}.$$

[1] When we cover Gaussian processes, we will have an infinite number of basis functions and still be able to make sensible predictions!

No held-out validation set is required.

Fitting a small number of parameters (σ_w and σ_y) to the marginal likelihood is less prone to overfitting than fitting everything (σ_w, σ_y, and $\mathbf{w}$) to the joint likelihood. However, overfitting is still possible.

Check your understanding

I run the computer program to random roll either a 10- or 10^6-sided "dice", as described early on in the note. The program outputs a `5`, what is the posterior probability of `sides=10` rather than `sides=1e6`?

Can you work out how to optimize the marginal likelihood $p(\mathcal{D}\,|\,\sigma_w, \sigma_y)$ for a linear regression model? Look back at the initial note on Bayesian regression for results that could be useful. In that note we were assuming that the hyperparameters σ_w and σ_y were known and fixed. In the notation of that note, the marginal likelihood would simply be $p(\mathcal{D})$, because we didn't bother to condition every expression on the hyperparameters. More guidance in the footnote[2].

Further Reading

Murphy 7.6.4 is on Bayesian model selection. For keen students, earlier sections give mathematical detail for a Bayesian treatment of the noise variance.

For keen students: Chapter 28 of MacKay's book has a lengthier discussion of Bayesian model comparison. Some time ago, I wrote a note discussing one of the figures in that chapter.

For very keen students: It can be difficult to put sensible priors on models with many parameters. In these situations it can sometimes be better to start out with a model class that we know is too simple, and only swap to a complex model when we have a lot of data. Bayesian model comparison can fail to tell us the best time to switch to a more complex model. The paper Catching up faster by switching sooner (Erven et al., 2012) has a nice language modelling example, and my thoughts are in the discussion of the paper.

For very keen students, Gelman et al.'s *Bayesian Data Analysis* book is a good starting point for reading about model checking and criticism. All models are wrong, but we want to improve parts of a model that are most strongly in disagreement with the data.

[2] Bayes' rule tells us that: $p(\mathbf{w}\,|\,\mathcal{D}) = p(\mathbf{w})\,p(\mathcal{D}\,|\,\mathbf{w})\,/\,p(\mathcal{D})$. The Bayesian regression note identified all of the distributions in this equation except for $p(\mathcal{D})$, so we can simply rearrange it to write $p(\mathcal{D})$ as a fraction containing three Gaussian distributions. We could optimize by grid search, or take the log of the expression and optimize with gradient-based methods.