Bayesian logistic regression and Laplace approximations

So far we have only performed Bayesian inference in two particularly tractable situations: 1) a small discrete problem (the card game); and 2) "linear-Gaussian models", where the observations are linear combinations of variables with Gaussian beliefs, to which we add Gaussian noise.

For most models, we cannot compute the equations for Bayesian prediction exactly. Logistic regression will be our working example. We'll look at how Bayesian predictions differ from regularized maximum likelihood. Then we'll look at approximate computation of the integrals, starting with the Laplace approximation.

Logistic regression

As a quick review, the logistic regression model gives the probability of a binary label given a feature vector:

$$P(y=1 \mid \mathbf{x}, \mathbf{w}) \;=\; \sigma(\mathbf{w}^\top \mathbf{x}) \;=\; 1/(1+e^{-\mathbf{w}^\top \mathbf{x}}).$$

We usually add a bias parameter b to the model, making the probability $\sigma(\mathbf{w}^\top \mathbf{x}+b)$. Although the bias is often dropped from the presentation, to reduce clutter. We can always work out how to add a bias back in, by including a constant element in the input features $\mathbf{x}$.

You'll see various notations used for the training data $\mathcal{D}$. The model gives the probability of a vector of outcomes $\mathbf{y}$ associated with a matrix of inputs X (where the nth row is $\mathbf{x}^{(n)\top}$). Maximum likelihood fitting maximizes the probability:

$$P(\mathbf{y} \mid X, \mathbf{w}) = \prod_n \sigma(z^{(n)} \mathbf{w}^\top \mathbf{x}^{(n)}), \qquad \text{where } z^{(n)} = 2y^{(n)} - 1, \;\; \text{if } \; y^{(n)} \in \{0, 1\}.$$

For compactness, we'll write this likelihood as $P(\mathcal{D} \mid \mathbf{w})$, even though really only the outputs $\mathbf{y}$ in the data are modelled. The inputs X are assumed fixed and known.

Logistic regression is most frequently fitted by a regularized form of maximum likelihood. For example L2 regularization fits an estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \left[\log P(\mathbf{y} \mid X, \mathbf{w}) - \lambda \mathbf{w}^\top \mathbf{w} \right].$$

We find a setting of the weights that make the training data appear probable, but discourage fitting extreme settings of the weights, that don't seem reasonable. Usually the bias weight will be omitted from the regularization term.

Just as with simple linear regression, we can instead follow a Bayesian approach. The weights are unknown, so predictions are made considering all possible settings, weighted by how plausible they are given the training data.

Bayesian logistic regression

The posterior distribution over the weights is given by Bayes' rule:

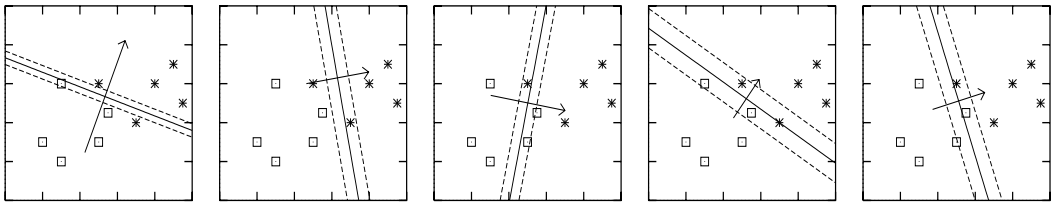
$$p(\mathbf{w} \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \mathbf{w})\, p(\mathbf{w})}{P(\mathcal{D})} \propto P(\mathcal{D} \mid \mathbf{w})\, p(\mathbf{w}).$$

The normalizing constant is the integral required to make the posterior distribution integrate to one:

$$P(\mathcal{D}) = \int P(\mathcal{D} \mid \mathbf{w})\, p(\mathbf{w}) \, \mathrm{d}\mathbf{w}.$$

The figures below[1] are for five different plausible sets of parameters, sampled from the posterior $p(\mathbf{w} \mid \mathcal{D})$.[2] Each figure shows the decision boundary $\sigma(\mathbf{w}^\top \mathbf{x}) = 0.5$ for one parameter vector as a solid line, and two other contours given by $\mathbf{w}^\top \mathbf{x} = \pm 1$.

1. The two figures in this section are extracts from Figure 41.7 of MacKay's textbook (p499). Murphy's Figures 8.5 and 8.6 contain a similar illustration.
2. It's not obvious how to generate samples from $p(\mathbf{w} \mid \mathcal{D})$, and in fact it's hard to do exactly. These samples were drawn approximately with a "Markov chain Monte Carlo" method.



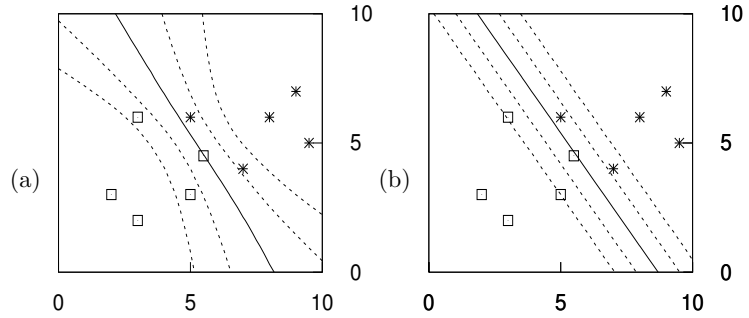
The axes in the figures above are the two input features x_1 and x_2. The model included a bias parameter, and the model parameters were sampled from the posterior distribution given data from the two classes as illustrated. The arrow, perpendicular to the decision boundary, illustrates the direction and magnitude of the weight vector.

Assuming that the data are well-modelled by logistic regression, it's clear that we don't know what the correct parameters are. That is, we don't know what parameters we would fit after seeing substantially more data. The predictions given the different plausible weight vectors differ substantially.

The Bayesian way to proceed is to use probability theory to derive an expression for the prediction we want to make:

$$\begin{aligned} P(y \mid \mathbf{x}, \mathcal{D}) &= \int p(y, \mathbf{w} \mid \mathbf{x}, \mathcal{D}) \, \mathrm{d}\mathbf{w} \\ &= \int P(y \mid \mathbf{x}, \mathbf{w}) \, p(\mathbf{w} \mid \mathcal{D}) \, \mathrm{d}\mathbf{w}. \end{aligned}$$

That is, we should average the predictive distributions $P(y \mid \mathbf{x}, \mathbf{w})$ for different parameters, weighted by how plausible those parameters are, $p(\mathbf{w} \mid \mathcal{D})$. Contours of this predictive distribution, $P(y=1 \mid \mathbf{x}, \mathcal{D}) \in \{0.5, 0.27, 0.73, 0.12, 0.88\}$, are illustrated in the left panel below. Predictions at some constant distance away from the decision boundary are less certain when further away from the training inputs. That's because the different predictors above disagreed in regions far from the data.



Again, the axes are the input features x_1 and x_2. The right hand figure shows $P(y=1 \mid \mathbf{x}, \mathbf{w}^*)$ for some fitted weights $\mathbf{w}^*$. No matter how these fitted weights are chosen, the contours have to be linear. The parallel contours mean that the uncertainty of predictions falls at the same rate when moving away from the decision boundary, no matter how far we are from the training inputs.

It's common to describe L2 regularized logistic regression as MAP (Maximum a posteriori) estimation with a Gaussian $\mathcal{N}(0, \sigma_w^2 \mathbb{I})$ prior on the weights. The "most probable"[3] weights, coincide with an L2 regularized estimate:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \left[\log p(\mathbf{w} \mid \mathcal{D})\right] = \arg\max_{\mathbf{w}} \left[\log P(\mathcal{D} \mid \mathbf{w}) - \frac{1}{2\sigma_w^2} \mathbf{w}^\top \mathbf{w}\right].$$

MAP estimation is *not* a "Bayesian" procedure. The rules of probability theory don't tell us to fix an unknown parameter vector to an estimate. We could view MAP as an approximation

3. "Most probable" is problematic for real-valued parameters. Really we are picking the weights with the highest probability density. But those weights aren't well-defined, because if we consider a non-linear reparameterization of the weights, the maximum of the pdf will be in a different place. That's why I prefer to describe estimating the weights as "regularized maximum likelihood" or "penalized maximum likelihood" rather than MAP.

to the Bayesian procedure, but the figure above illustrates that it is a crude one: the Bayesian predictions (left) are qualitatively different to the MAP ones (right).

Unfortunately, we can't evaluate the integral for predictions $P(y \mid \mathbf{x}, \mathcal{D})$ in closed form. Making model choices for Bayesian logistic regression is also computationally challenging. The marginal probability of the data, $P(\mathcal{D})$, is the marginal likelihood of the model, which we might write as $P(\mathcal{D} \mid \mathcal{M})$ when we are evaluating some model choices $\mathcal{M}$ (such as basis functions and hyperparameters). We also can't evaluate the integral for $P(\mathcal{D})$ in closed form.

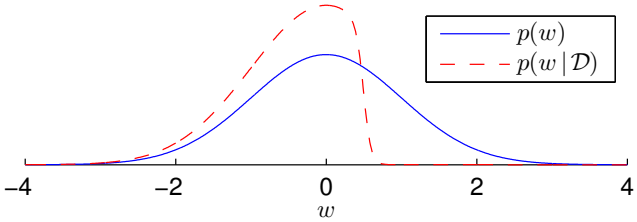
The logistic regression posterior is sometimes approximately Gaussian

We're able to do some integrals involving Gaussian distributions. The posterior distribution over the weights $p(\mathbf{w} \mid \mathcal{D})$ is *not* Gaussian, but we can make progress if we can approximate it with a Gaussian.

First, I've contrived an example to illustrate how the posterior over the weights can look non-Gaussian. We have a Gaussian prior with one sigmoidal likelihood term. Here we assume we know the bias[4] is 10, and we have one datapoint with $y=1$ at $x=-20$:

$$\begin{aligned} p(w) &\propto \mathcal{N}(w; 0, 1) \\ p(w \mid \mathcal{D}) &\propto \mathcal{N}(w; 0, 1)\, \sigma(10 - 20w). \end{aligned}$$

We are now fairly sure that the weight isn't a large positive value, because otherwise we'd have probably seen $y=0$. We (softly) slice off the positive region[5] and renormalize to get the posterior distribution illustrated below:

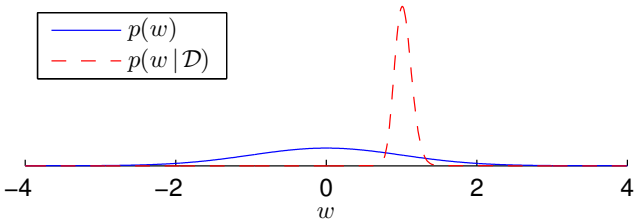


The distribution is asymmetric and so clearly not Gaussian. Every time we multiply the posterior by a sigmoidal likelihood, we softly carve away half of the weight space in some direction. While the posterior distribution has no neat analytical form, the distribution over plausible weights often does look Gaussian after many observations.

As another example, I generated $N=500$ labels, $\{z^{(n)}\}$, from a logistic regression model with no bias and with $w=1$ at $x^{(n)} \sim \mathcal{N}(0, 10^2)$. Then,

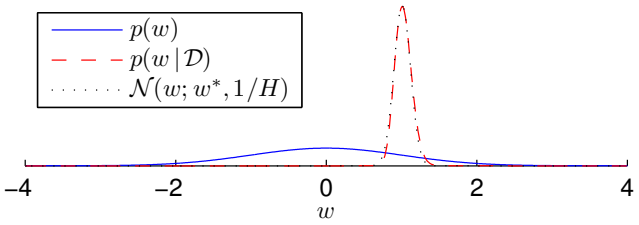
$$\begin{aligned} p(w) &\propto \mathcal{N}(w; 0, 1) \\ p(w \mid \mathcal{D}) &\propto \mathcal{N}(w; 0, 1) \prod_{n=1}^{500} \sigma(w x^{(n)} z^{(n)}), \qquad z^{(n)} \in \{\pm 1\}. \end{aligned}$$

The posterior now appears to be a beautiful bell-shape:



Fitting a Gaussian distribution (using the Laplace approximation, next section) shows that the distribution isn't quite Gaussian… but it's close:

4. Perhaps we have many datapoints and have fitted the bias precisely, but we have one datapoint that has a novel feature turned on, and the example is showing the posterior over the weight that interacts with that one feature.
5. If it's not obvious what's going on, plot $\sigma(10 - 20w)$ against w. We are multiplying our prior by this soft step function, which multiplies the prior by nearly one on the left, and nearly zero on the right.



The Laplace Approximation

There are multiple ways that we could try to fit a distribution with a Gaussian form. For example, we could try to match the mean and variance of the distribution. The Laplace approximation is another possible way to approximate a distribution with a Gaussian. It can be seen as an incremental improvement of the MAP approximation to Bayesian inference, and only requires some additional derivative computations.

We can only evaluate the posterior distribution up to a constant: we can evaluate the joint probability $p(\mathbf{w}, \mathcal{D})$, but not the normalizer $P(\mathcal{D})$. We match the shape of the posterior using $p(\mathbf{w}, \mathcal{D})$, and then the approximation can be used to approximate $P(\mathcal{D})$.

The Laplace approximation sets the mode of the Gaussian approximation to the mode of the posterior distribution, and matches the curvature of the log probability density at that location. We need to be able to evaluate first and second derivatives of $\log P(\mathbf{w}, \mathcal{D})$.

The rest of the notes just fills in the details. I'm not adding much to MacKay's textbook pp341–342, or Murphy's book p255. Although I try to go slightly more slowly and show some pictures of what can go wrong. A concrete example is on Tutorial sheet 7.

Matching the distributions

First of all we find the most probable setting of the parameters:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \; p(\mathbf{w} \mid \mathcal{D}) = \arg\max_{\mathbf{w}} \; \log p(\mathbf{w}, \mathcal{D}).$$

The conditional probability on the left is what we intuitively want to optimize. The maximization on the right gives the same answer, but contains the term we will actually compute. Reminder: why do we take the log?[6]

We usually find the mode of the distribution by minimizing an 'energy', which is the negative log-probability of the distribution up to a constant. For a posterior distribution, we can define the energy as:

$$E(\mathbf{w}) = -\log p(\mathbf{w}, \mathcal{D}), \qquad \mathbf{w}^* = \arg\min_{\mathbf{w}} E(\mathbf{w}).$$

We minimize it as usual, using a gradient-based numerical optimizer.

The minimum of the energy is a turning point. For a scalar variable w the first derivative $\frac{\partial E}{\partial w}$ is zero and the second derivative gives the curvature of this turning point:

$$H = \left. \frac{\partial^2 E(w)}{\partial w^2} \right|_{w=w^*}.$$

The notation means that we evaluate the second derivative at the optimum, $w = w^*$. If H is large, the slope (the first derivative) changes rapidly from a steep descent to a steep ascent. We should approximate the distribution with a narrow Gaussian. Generalizing to multiple

6. Because log is a monotonic transformation, maximizing the log of a function is equivalent to maximizing the original function. Often the log of a distribution is more convenient to work with, less prone to numerical problems, and closer to an ideal quadratic function that optimizers like.

variables $\mathbf{w}$, we know $\nabla_{\mathbf{w}} E$ is zero at the optimum and we evaluate the *Hessian*, a matrix with elements:

$$H_{ij} = \left. \frac{\partial^2 E(\mathbf{w})}{\partial w_i \partial w_j} \right|_{\mathbf{w}=\mathbf{w}^*}.$$

This matrix tells us how sharply the distribution is peaked in different directions.

For comparison, we can find the optimum and curvature that we would get if our distribution were Gaussian. For a one-dimensional distribution, $\mathcal{N}(\mu, \sigma^2)$, the energy (the negative log-probability up to a constant) is:

$$E_{\mathcal{N}}(w) = \frac{(w-\mu)^2}{2\sigma^2}.$$

The minimum is $w^* = \mu$, and the second derivative $H = 1/\sigma^2$, implying the variance is $\sigma^2 = 1/H$. Generalizing to higher dimensions, for a Gaussian $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$, the energy is:

$$E_{\mathcal{N}}(\mathbf{w}) = \frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^\top \Sigma^{-1} (\mathbf{w}-\boldsymbol{\mu}),$$

with $\mathbf{w}^* = \boldsymbol{\mu}$ and $H = \Sigma^{-1}$, implying the covariance is $\Sigma = H^{-1}$.

Therefore matching the minimum and curvature of the 'energy' (negative log-probability) to those of a Gaussian energy gives the Laplace approximation to the posterior distribution:

$$\boxed{p(\mathbf{w} \,|\, \mathcal{D}) \approx \mathcal{N}(\mathbf{w};\, \mathbf{w}^*, H^{-1})}$$

Approximating the normalizer Z

Evaluating our approximation for a D-dimensional distribution gives:

$$p(\mathbf{w} \,|\, \mathcal{D}) = \frac{p(\mathbf{w}, \mathcal{D})}{P(\mathcal{D})} \approx \mathcal{N}(\mathbf{w};\, \mathbf{w}^*, H^{-1}) = \frac{|H|^{1/2}}{(2\pi)^{D/2}} \exp\left(-\frac{1}{2}(\mathbf{w}-\mathbf{w}^*)^\top H (\mathbf{w}-\mathbf{w}^*) \right).$$

At the mode $\mathbf{w}^* = \mathbf{w}$, the exponential term disappears and we get:

$$\frac{p(\mathbf{w}^*, \mathcal{D})}{P(\mathcal{D})} \approx \frac{|H|^{1/2}}{(2\pi)^{D/2}}, \qquad \boxed{P(\mathcal{D}) \approx \frac{p(\mathbf{w}^*, \mathcal{D})(2\pi)^{D/2}}{|H|^{1/2}}}.$$

An equivalent expression is

$$\boxed{P(\mathcal{D}) \approx p(\mathbf{w}^*, \mathcal{D})\, |2\pi H^{-1}|^{1/2},}$$

where $|\cdot|$ means take the determinant of the matrix.

When some people say "the Laplace approximation", they are referring to this approximation of the normalization $P(\mathcal{D})$, rather than the intermediate Gaussian approximation to the distribution.

Is the approximation reasonable?

If we think that the Energy is well-behaved and sharply peaked around the mode of the distribution, we might think that we can approximate it with a Taylor series. In one dimension we write

$$\begin{aligned} E(w^* + \delta) &\approx E(w^*) \;+\; \left. \frac{\partial E}{\partial w} \right|_{w^*} \delta \;+\; \frac{1}{2} \left. \frac{\partial^2 E}{\partial w^2} \right|_{w^*} \delta^2 \\ &\approx E(w^*) \;+\; \frac{1}{2} H \delta^2, \end{aligned}$$

where the second term disappears because $\frac{\partial E}{\partial w}$ is zero at the optimum. In multiple dimensions this Taylor approximation generalizes to:

$$E(\mathbf{w}^* + \boldsymbol{\delta}) \approx E(\mathbf{w}^*) + \tfrac{1}{2}\boldsymbol{\delta}^\top H \boldsymbol{\delta}.$$

A quadratic energy (negative log-probability) implies a Gaussian distribution. The distribution is close to the Gaussian fit when the Taylor series is accurate.

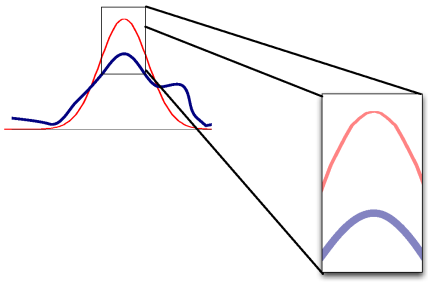
For models with a fixed number of identifiable parameters, the posterior becomes tightly peaked in the limit of large datasets. Then the Taylor expansion of the log-posterior doesn't need to be extrapolated far and will be accurate. Search term for more information: "Bayesian central limit theorem".

The Laplace approximation doesn't always work well!

Despite the theory above, it is easy for the Laplace approximation to go wrong.

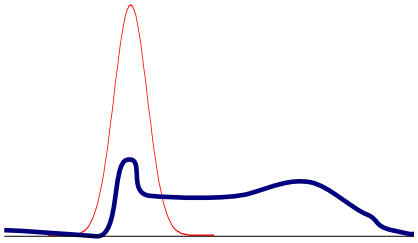
In high dimensions, there are many directions in parameter space where there might only be a small number of informative datapoints. Then the posterior could look like the first asymmetrical example in this note.

If the mode and curvature are matched, but the distribution is otherwise non-Gaussian, then the value of the densities won't match[7].



As a result, the approximation of $P(\mathcal{D})$ will be poor.

One way for a distribution to be non-Gaussian is to be multi-modal. The posterior of logistic regression only has one mode, but the posterior for neural networks will be multimodal. Even if capturing one mode is reasonable, an optimizer could get stuck in bad local optima.



In models with many parameters, the posterior will often be flat in some direction, where parameters trade off each other to give similar predictions. When there is zero curvature in some direction, the Hessian isn't positive definite and we can't get a meaningful approximation.

Further Reading

Suggested reading: Murphy Section 8.4 to 8.4.4 inclusive. You can skip 8.4.2 on BIC.

Similar material is covered by MacKay, Ch. 41, pp492–503, and Ch. 27, pp341–342. (Section 41.4 uses non-examinable methods — skim over on first reading.)

7. The final two figures in this note come from previous MLPR course notes, by one of Amos Storkey, Chris Williams, or Charles Sutton.

The Laplace approximation was used in some of the earliest Bayesian neural networks although — as presented here — it's now rarely used. However, the idea does occur in recent work, such as on continual learning (Kirkpatrick et al., Google Deepmind, 2017) and a more sophisticated variant is used by the popular statistical package, R-INLA.