Bayesian regression

We now revisit regression, fitting real-valued outputs, with a more elaborate statistical approach than before. Previously, we fitted a scalar function, which represented a best guess of an output at a location. Now we will represent a whole probability distribution over possible outputs for each input location. Given this probabilistic model, we can use "Bayesian" reasoning (probability theory) to make predictions. We won't need to cross-validate as many choices, and we will be able to specify how uncertain we are about the model's parameters and its predictions. We won't achieve these advantages in this note, but that's the motivation for what we're setting up here.

A probabilistic model for regression

When we created classifiers, it was useful to model the probability distribution over possible labels at each position in input space $P(y \mid \mathbf{x})$. We can also write down a probabilistic model for regression models. The simplest starting point is a Gaussian model:

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

where $f(\mathbf{x}; \mathbf{w})$ is any function specified by parameters $\mathbf{w}$. For example, the function f could be specified by a linear model, a linear model with basis functions, or a neural network. We have explicitly written down that we believe the differences between observed outputs and underlying function values should be Gaussian distributed with variance σ_y^2.[1]

The starting point for estimating most probabilistic models is "maximum likelihood", finding the parameters that make the data most probable under the model. Numerically, we minimize the negative log likelihood:[2]

$$\begin{aligned} -\log p(\mathbf{y} \mid X, \mathbf{w}) &= -\sum_n \log p(y^{(n)} \mid \mathbf{x}^{(n)}, \mathbf{w}) \\ &= \frac{1}{2\sigma_y^2} \sum_{n=1}^{N} \left[(y^{(n)} - f(\mathbf{x}^{(n)}; \mathbf{w}))^2 \right] + \frac{N}{2} \log(2\pi\sigma_y^2). \end{aligned}$$

If we assume the noise variance σ_y^2 is constant, we fit the parameters $\mathbf{w}$ simply by minimizing square error, exactly as we have done earlier in the course.

By following this probabilistic interpretation, we are making more explicit assumptions than necessary to justify least squares[3]. However, adopting a probabilistic framework can make it easier to explore and reason about different models. If our measuring instrument reported different noise variances $(\sigma_y^{(n)})^2$ for different observed $y^{(n)}$ values, we could immediately modify the likelihood and obtain a sensible new cost function. Similarly, we could build a more robust model (as we did for classification) by writing down a noise model with heavier tails to better model outliers. Again, we would immediately obtain a new cost function.

For now we will stay with a simple linear regression model, with known constant noise variance, and look at how to reason about its unknown weight parameters.

Representing uncertainty with distributions

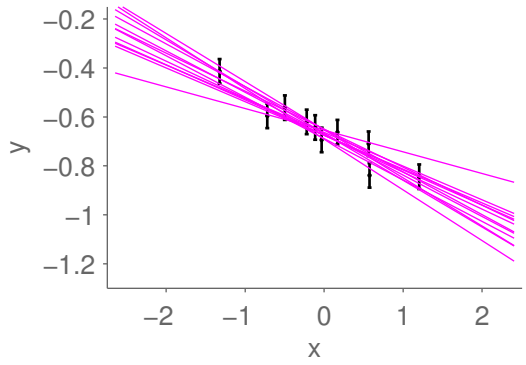
[If this section doesn't make sense, just keep reading. It's just meant as motivation, and the next section is more concrete. Most people need to see a few restatements of Bayesian learning before it makes sense.]

---

1. Notation: In the following notes we will have variances of multiple quantities, so I need to start labelling them. I use σ_y^2 for the conditional variance of an observation y given the underlying function value f. It is not the variance of the marginal distribution $p(y)$. I will use this notation again when covering Gaussian processes (GPs). The Murphy textbook also uses σ_y^2 when covering GPs, but plain σ^2 for the noise variance for Bayesian linear regression.
2. In this note, $\mathbf{y}$ is an $N \times 1$ vector containing a scalar observation for each training example.
3. e.g., `https://en.wikipedia.org/wiki/Gauss%E2%80%93Markov_theorem`

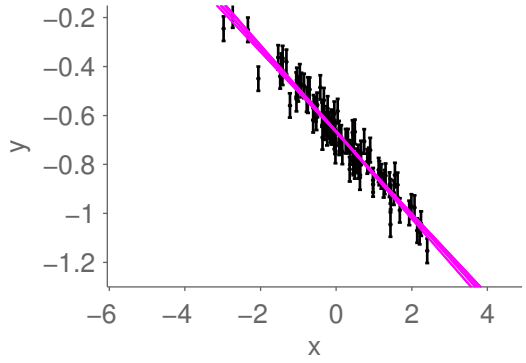
As a physics undergraduate I plotted experimental results, with error bars, in a paper lab book. We were instructed to find a line of best fit by eye, and draw it using a pencil and ruler. We'd also draw the steepest and shallowest straight lines that were still reasonably good fits to the data, to get a sense for how uncertain the original fit was. Given $\pm\sigma_y$ error bars on our observed y values, we wanted each fitted line to go through about $2/3$ of the error bars. But there are many ways to draw such a line: with limited noisy observations, we can't know where an underlying function is.

What I've done in the figure below is slightly different. Rather than just showing extreme lines, I've drawn a selection of twelve lines, each of which could plausibly have generated the data. If I hid the line that I actually used to generate the data in this collection, you would have trouble identifying it with certainty.



One of the lines does look somewhat less plausible than the others. But it could be close to the true line: maybe the left-most observation happened to have a large amount of noise added to it, and the line is less steep than you think.

If we observe a lot more points, we can be more certain where the underlying line is. The range of plausible lines we might draw should become more limited, perhaps like this:



I've drawn the lines almost on top of each other, so they appear as one thick line. I am fairly certain that the true function is there.

If we wanted an intelligent agent to draw these plausible lines of fit for us, how should it do it? Or more generally, how can we systematically express the uncertainty that we should have in conclusions based on limited and noisy data?

One formal approach to handling uncertainty is to use probability distributions to represent our beliefs, and probability theory to update them.[4]

In the context of line fitting, beliefs are represented by a probability distribution over the parameters. When we are fairly certain, the distribution over parameters should have low variance, concentrating its probability mass on a small region of parameter space. Given only a few datapoints, our distribution over plausible parameters should be much broader.

4. For keen students: a justification of this view is outlined in "Probability, frequency and reasonable expectation", R. T. Cox, *American Journal of Physics*, 14(1):1–13, 1946. It's a readable paper.

In the figures above, I drew samples from probability distributions over parameters, and plotted the resulting functions. The lines show just twelve of the many reasonable explanations of the data as represented by a distribution over the model parameters. I got the distributions from Bayes' rule as described in the next section.

Bayes' rule for beliefs about parameters

To apply probability theory to obtain beliefs we need both a probabilistic model, and *prior beliefs* about that model.

Our example model for a straight line fit is:

$$f(x; \mathbf{w}) = w_1 x + w_2, \qquad p(y \mid x, \mathbf{w}) = \mathcal{N}(y;\, f(x; \mathbf{w}), \sigma_y^2).$$

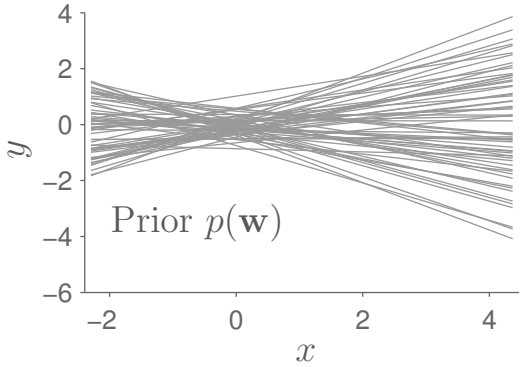
If we knew the parameters $\mathbf{w}$, we could compute the function for any x and simulate what typical observations at that location would look like. However, when learning from data, we don't know the parameters $\mathbf{w}$.

The prior

Our *prior beliefs* are represented by a distribution over the parameters, specifying which models we think are plausible before observing any data. For example, if we set:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w};\, \mathbf{0}, 0.4^2 \mathbb{I}),$$

we think the following functions are all reasonable models we could consider:



Each of the lines above is a function $f(x; \mathbf{w})$ where I have sampled the parameters $\mathbf{w}$ from the prior distribution. They are quite bunched up around the origin. If I wanted to consider larger intercepts at $x = 0$, I could pick a broader prior on the bias parameter w_2.

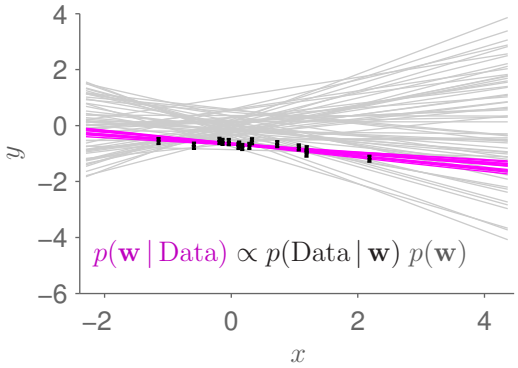
The posterior

The *posterior distribution* gives the beliefs about the model parameters that we should have *after* observing data $\mathcal{D} = \{\mathbf{x}^{(n)}, y^{(n)}\}$. These beliefs are represented by the distribution $p(\mathbf{w} \mid \mathcal{D})$, pronounced "p of $\mathbf{w}$ *given* $\mathcal{D}$", and are obtained by Bayes' rule.

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

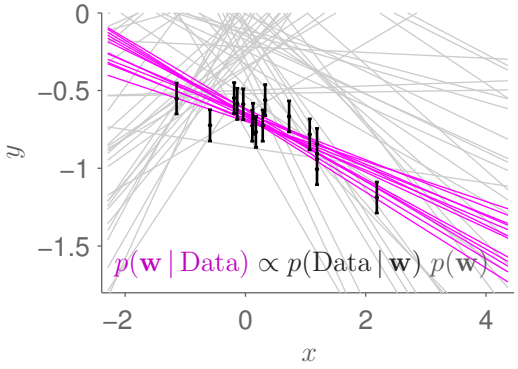
Bayes' rule reweights or updates the prior distribution, so that parameters that are more compatible with the data become more probable.

Applying Bayes rule to our toy regression problem (skipping the maths of *how* to do it for the moment) gives beliefs centred around models that fit the observed data-points. Sampling from that distribution gives the plot below:



The light gray lines show the samples from the prior distribution that we had before. The small black bars show data points (with error bars). The magenta lines show 12 samples from the posterior distribution. As is usually the case (if our data is useful), the posterior distribution concentrates in a much smaller region of parameter space than the prior. The corresponding straight-line functions are also less spread out. Under the posterior, we only believe that lines passing close to the data are plausible.

Zooming in to the same diagram, we can get a better view of the data and the possible models sampled from the posterior distribution:



The plausible lines sampled from the posterior distribution naturally spread out as we move away from the observed data. The statistical machinery automatically tells us that we are less certain about where the function is as we move away from where we have measured it.

Computing the Posterior

For general models, the posterior distribution won't have a convenient form that we can represent with a few parameters. The likelihood is a product of N terms for N datapoints, and that product could potentially create a complicated function.

A *conjugate prior* is one where the product of the prior and likelihood combines to give a distribution with the same functional form as the prior. A Gaussian prior over linear regression weights is conjugate to its linear-Gaussian likelihood, and we obtain a Gaussian posterior:

$$\begin{aligned} p(\mathbf{w} \,|\, \mathcal{D}) &\propto p(\mathbf{w})\, p(\mathcal{D} \,|\, \mathbf{w}), \\ &\propto \mathcal{N}(\mathbf{w};\, \mathbf{0},\, \sigma_w^2 \mathbb{I}) \prod_n \mathcal{N}(y^{(n)}; \mathbf{w}^\top \boldsymbol{\phi}(x^{(n)}),\, \sigma_y^2), \\ &\propto \mathcal{N}(\mathbf{w};\, \mathbf{0},\, \sigma_w^2 \mathbb{I})\, \mathcal{N}(\mathbf{y};\, \Phi\mathbf{w},\, \sigma_y^2 \mathbb{I}), \end{aligned}$$

where $\boldsymbol{\phi}(\mathbf{x}) = [x \;\; 1]^\top$ is an augmented input vector, and Φ is a matrix containing these augmented vectors in each row. Here I've assumed a zero-mean prior, and a spherical prior covariance. In Murphy's notation for the prior parameters: $\mathbf{w}_0 = \mathbf{0}$ and $V_0 = \sigma_w^2 \mathbb{I}$, which set our beliefs after observing 0 datapoints. We could set the prior mean $\mathbf{w}_0$ and covariance V_0 to other values, for example to allow the "bias" parameter that specifies the intercept to vary more.

The posterior is proportional to the exponential of a quadratic in $\mathbf{w}$, which means it is a Gaussian distribution. Murphy writes the answer in the form: $p(\mathbf{w} \mid \mathcal{D}) = \mathcal{N}(\mathbf{w}; \mathbf{w}_N, V_N)$, the beliefs after observing N datapoints. As in a tutorial exercise, we can identify the posterior mean $\mathbf{w}_N$ and covariance V_N from the linear and quadratic coefficients of $\mathbf{w}$ inside the exponential. It's just linear algebra grunt work, which results in:

$$V_N = \sigma_y^2(\sigma_y^2 V_0^{-1} + \Phi^\top \Phi)^{-1}$$
$$\mathbf{w}_N = V_N V_0^{-1}\mathbf{w}_0 + \frac{1}{\sigma_y^2} V_N \Phi^\top \mathbf{y}.$$

Or replace Φ with X if you haven't applied a basis function transformation. I am not expecting you to memorize these expressions!

Recommended reading

Murphy Chapter 7 introduces linear regression with a probabilistic perspective from the beginning. Bayesian linear regression is in Section 7.6. There is a demo in Figure 7.11 that comes with code.

Possible exercises

Set a prior over parameters that lets the intercept of the function vary more, while maintaining the same distribution over slopes as in the demonstration in this note. Plot the straight line functions corresponding to some parameter vectors sampled from your new prior.

If you want to work through the maths and implement a Bayesian inference problem, the following work-up exercise may test if you understand how to go about it. Assume a Gaussian prior over a single number m. Sample a value for m from this prior and generate $N = 12$ datapoints from $\mathcal{N}(m, 1)$. What should your posterior be given these observations? You could run a series of trials where you draw a value for m from the prior, simulate data and compute a posterior. How often is the true value of m within $\pm$ one standard deviation of your posterior mean?

Further detail

Definitely non-examinable: I first learned Bayesian methods from David MacKay. Chapter 3 of his book, describes the undergraduate physics question where he first learned about Bayes' rule for inferring an unknown parameter. Depending on your background, you may find that story interesting.

Historical note: Using probabilities to describe beliefs has been somewhat controversial over time. There's no controversy in Bayes' rule: it's a rule of probability theory, a direct consequence of the product rule. It's using Bayes' rule to obtain beliefs, rather than frequencies, that has been controversial.