Linear regression

We are ready to consider our first machine-learning problem: *linear regression.* Suppose that we are interested in the values of a function $y(\mathbf{x})\colon \mathbb{R}^d \to \mathbb{R}$, where $\mathbf{x}$ is a d-dimensional vector-valued input. We assume that we have made some observations of this mapping, $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, to serve as training data. Given these examples, the goal of regression is to be able to predict the value of y at a new input location $\mathbf{x}_*$. In other words, we want to learn from $\mathcal{D}$ a (hopefully accurate!) prediction function (also called a *regression function*) $\hat{y}(\mathbf{x}_*)$.

Without any further restrictions on $\hat{y}$, the problem is not well-posed. In general, I could choose any function $\hat{y}$. How do I choose a good function? In particular, how do I effectively use the training observations $\mathcal{D}$ to guide this choice?

The space of all possible regression functions $\hat{y}$ is enormous; to restrict this space and make learning more tractable, *linear regression* assumes the relationship between $\mathbf{x}$ and y is "mostly" linear:

$$y(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} + \varepsilon(\mathbf{x}), \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^d$ is a vector of parameters, and $\varepsilon(\mathbf{x})$ represents a departure from the underlying linear function $\mathbf{x}^\top \mathbf{w}$ at $\mathbf{x}$. The value of $\varepsilon(\mathbf{x})$ is also called the *residual.* If the underlying linear trend closely matches the data, we can expect small residuals.

Note that this model does not explicitly contain an intercept term. Instead, we normally add a constant feature to the input vector $\mathbf{x}$ as in $\mathbf{x}' = [1, \mathbf{x}]^\top$. In this case, the value w_1 may be seen as an intercept term, allowing the hyperplane to avoid passing through the origin at $\mathbf{x} = 0$. To avoid messy notation, we will not explicitly write the prime on the inputs and rather assume this expansion has already been done.

In general, we can imagine applying any function to $\mathbf{x}$ to serve as the inputs to our linear regression model. For example, to accomplish polynomial regression of a given degree k, we might take $\mathbf{x}' = [1, \mathbf{x}, \mathbf{x}^2, \ldots, \mathbf{x}^k]^\top$, where exponentiation is pointwise. A transformation of this type may be summarized by a map $\mathbf{x} \mapsto \phi(\mathbf{x})$; this is known as a *basis expansion.* With a nonlinear basis expansion, we can learn nonlinear regression functions. We assume in the following that such an expansion, if desired, has already been applied to the inputs.

Let us collect the N training examples into a matrix of training inputs $\mathbf{X} \in \mathbb{R}^{N \times d}$ and a vector of associated outputs $\mathbf{y} \in \mathbb{R}^N$. Then our assumed linear relationship in (1), when applied to the training data, may be written

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon},$$

where we have also collected the residuals into the vector $\boldsymbol{\varepsilon}$.

There are many classical approaches for estimating the parameter vector $\mathbf{w}$; below, we describe two.

Ordinary least squares

As mentioned previously, a "good" linear mapping $\mathbf{x}^\top \mathbf{w}$ should result in small residuals. Our goal is to minimize the magnitude of the residuals at every possible test input $\mathbf{x}_*$. Of course, this is impossible, so we settle for the next best thing: we minimize the magnitude of the residuals on our training data.

The classical approach is to estimate $\mathbf{w}$ by minimizing the sum of the squared residuals on the

training data:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{i=1}^{N} (\mathbf{x}_i^\top \mathbf{w} - y_i)^2. \tag{2}$$

This approach is called *ordinary least squares*. The underlying assumption is that the magnitude of the residuals are all independent from each other, so that we can simply sum up the squared residuals and minimize. (If the residuals were instead correlated, we would have to consider the interaction between pairs of residuals. This is also possible and gives rise to *generalized least squares*.)

It turns out that the minimization problem in (2) has an exact, closed-form solution:

$$\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

To predict the value of y at a new input $\mathbf{x}_*$, we plug our estimate of the parameter vector into (1):

$$\hat{y}(\mathbf{x}_*) = \mathbf{x}_*^\top \hat{\mathbf{w}} = \mathbf{x}_*^\top (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Ordinary least squares is by far the most commonly applied linear regression procedure.

Ridge regression

Occasionally, the ordinary least squares approach can lead to problems. In particular, when the training data lie extremely close to a particular hyperplane (or when the number of observations is less than the dimension, and we may find a hyperplane intersecting the data exactly), the magnitudes of the estimated parameters $\hat{\mathbf{w}}$ can become ill-behaved and extremely large. This is an example of *overfitting*.

A priori, we might reasonably expect or desire the values of $\mathbf{w}$ to be relatively small. For this reason, so-called *penalization* or *regularization methods* modify the objective in (2) to avoid such pathologies. The most common approach is to add a term to the objective encouraging small entries of $\mathbf{w}$, for example

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{i=1}^{N} (\mathbf{x}_i^\top \mathbf{w} - y_i)^2 + \lambda \|\mathbf{w}\|_2^2, \tag{3}$$

where we have added the squared 2-norm of $\mathbf{w}$ to the function to be minimized, thereby penalizing large-magnitude entries of $\mathbf{w}$. The scalar $\lambda \geq 0$ serves to trade off the contributions of the residual term and the penalty term, and is a parameter of the model that we may tune as desired. As $\lambda \to 0$, we recover the ordinary least squares objective.

The minimization problem in (3) may also be solved exactly, giving the slightly modified estimator

$$\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Again, as $\lambda \to 0$ (corresponding to very small penalties on the magnitude of $\mathbf{w}$ entries), we see that the resulting estimator converges to the ordinary least squares estimator. As $\lambda \to \infty$ (corresponding to very large penalties on the magnitude of $\mathbf{w}$ entries), the estimator converges to $\mathbf{0}$.

This approach is known as *ridge regression,* named for the additional term $\lambda \mathbf{I}$ in the estimator, which when viewed from "above" can whimsically be described as resembling a mountain ridge.

Bayesian linear regression

Here we consider a Bayesian approach to linear regression. Consider again the assumed underlying linear relationship (1):

$$y(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} + \varepsilon(\mathbf{x}).$$

Here the vector $\mathbf{w}$ serves as the unknown parameter of the model, which we will reason about using the Bayesian method. For convenience, we will select a multivariate Gaussian prior distribution on $\mathbf{w}$:

$$p(\mathbf{w} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

As before, we might expect *a priori* that the entries of $\mathbf{w}$ are small. Furthermore, in the absence of any evidence otherwise, we might reason that all potential directions of $\mathbf{w}$ are equally likely. These two assumptions can be codified by choosing a multivariate Gaussian distribution for $\mathbf{w}$ with zero mean and isotropic diagonal covariance $s^2\mathbf{I}$:

$$p(\mathbf{w} \mid s^2) = \mathcal{N}(\mathbf{w}; \mathbf{0}, s^2\mathbf{I}). \tag{4}$$

Given our observed data $\mathcal{D} = (\mathbf{X}, \mathbf{y})$, we wish to form the posterior distribution $p(\mathbf{w} \mid \mathcal{D})$. We will do so by deriving the joint distribution between the weight vector $\mathbf{w}$ and the observed vector of values $\mathbf{y}$, given the inputs $\mathbf{X}$. We write again the assumed linear relationship in our training data:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}.$$

Given $\mathbf{w}$ and $\mathbf{X}$, the only uncertainty in the above is the additive residuals $\boldsymbol{\varepsilon}$. We assume that the residuals are independent, unbiased, and tend to be "small." We may accomplish this by modeling each entry of $\boldsymbol{\varepsilon}$ as arising from zero-mean, independent, identically distributed Gaussian noise with variance σ^2:

$$p(\boldsymbol{\varepsilon} \mid \sigma^2) = \mathcal{N}(\boldsymbol{\varepsilon}; \mathbf{0}, \sigma^2\mathbf{I}).$$

Define $\mathbf{f} = \mathbf{X}\mathbf{w}$, and note that $\mathbf{f}$ is a linear transformation of the multivariate-Gaussian distributed $\mathbf{w}$; therefore, we may easily derive its distribution:

$$p(\mathbf{f} \mid \mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{f}; \mathbf{X}\boldsymbol{\mu}, \mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top).$$

Now the observation vector $\mathbf{y}$ is a sum of two independent multivariate Gaussian distributed vectors $\mathbf{f}$ and $\boldsymbol{\varepsilon}$. Therefore, we may derive its prior distribution as well:

$$p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) = \mathcal{N}(\mathbf{y}; \mathbf{X}\boldsymbol{\mu}, \mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2\mathbf{I}).$$

Finally, we compute the covariance between $\mathbf{y}$ and $\mathbf{w}$:

$$\operatorname{cov}[\mathbf{y}, \mathbf{w}] = \operatorname{cov}[\mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}, \mathbf{w}] = \operatorname{cov}[\mathbf{X}\mathbf{w}, \mathbf{w}] = \mathbf{X}\operatorname{cov}[\mathbf{w}, \mathbf{w}] = \mathbf{X}\boldsymbol{\Sigma},$$

where we have used the linearity of covariance and the fact that the noise vector $\boldsymbol{\varepsilon}$ is independent of $\mathbf{w}$.

Now the joint distribution of $\mathbf{w}$ and $\mathbf{y}$ is multivariate Gaussian:

$$p\left(\begin{bmatrix}\mathbf{w}\\ \mathbf{y}\end{bmatrix} \mid \mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{w}\\ \mathbf{y}\end{bmatrix}; \begin{bmatrix}\boldsymbol{\mu}\\ \mathbf{X}\boldsymbol{\mu}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma} & \boldsymbol{\Sigma}\mathbf{X}^\top\\ \mathbf{X}\boldsymbol{\Sigma} & \mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2\mathbf{I}\end{bmatrix}\right).$$

We apply the formula for conditioning multivariate Gaussians on subvectors to derive the posterior distribution of $\mathbf{w}$ given the data $\mathcal{D}$:

$$p(\mathbf{w} \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}),$$

where

$$\begin{aligned}
\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}} &= \boldsymbol{\mu} + \boldsymbol{\Sigma}\mathbf{X}^\top(\mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\mu}); \\
\boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}} &= \boldsymbol{\Sigma} - \boldsymbol{\Sigma}\mathbf{X}^\top(\mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2\mathbf{I})^{-1}\mathbf{X}\boldsymbol{\Sigma}.
\end{aligned}$$

Making predictions

Suppose we wish to use our model to predict the outputs $\mathbf{y}_*$ associated with a set of inputs $\mathbf{X}_*$. Writing again $\mathbf{y}^* = \mathbf{X}_*\mathbf{w} + \boldsymbol{\varepsilon}$, we derive:

$$p(\mathbf{y}_* \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) = \mathcal{N}(\mathbf{y}_*; \mathbf{X}_*\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \mathbf{X}_*\boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}\mathbf{X}_*^\top + \sigma^2\mathbf{I}).$$

Therefore, we have a full, joint probability distribution over the outputs $\mathbf{y}_*$. If we must make point estimates, we choose a loss function for our predictions and use Bayesian decision theory. For example, to minimize expected squared loss, we predict the posterior mean.

An equivalent way to derive this result is to use the sum rule as follows:

$$\begin{aligned}
p(\mathbf{y}_* \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) &= \int p(\mathbf{y}_* \mid \mathbf{w}, \sigma^2) p(\mathbf{w} \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) \, \mathrm{d}\mathbf{w} \\
&= \int \mathcal{N}(\mathbf{y}_*; \mathbf{X}_*\mathbf{w}, \sigma^2\mathbf{I}) \, \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}) \, \mathrm{d}\mathbf{w} \\
&= \mathcal{N}(\mathbf{y}_*; \mathbf{X}_*\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \mathbf{X}_*\boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}\mathbf{X}_*^\top + \sigma^2\mathbf{I}),
\end{aligned}$$

where we have used a slightly more general form of the convolution formula for multivariate Gaussians. In this case, we view the prediciton process as considering the predictions we would make with every possible value of the parameters $\mathbf{w}$ and averaging them according to their plausibility.

Connection to ridge regression

For the simple prior (4), the posterior mean can be rewritten as:

$$\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}} = s^2\mathbf{X}^\top(s^2\mathbf{X}\mathbf{X}^\top + \sigma^2\mathbf{I})^{-1}\mathbf{y} = \left(\mathbf{X}^\top\mathbf{X} + \frac{\sigma^2}{s^2}\mathbf{I}\right)^{-1}\mathbf{X}^\top\mathbf{y},$$

where we have factored out the s^2 in the inverse and have applied the matrix identity

$$(\mathbf{A}\mathbf{B} + c\mathbf{I})^{-1}\mathbf{A} = \mathbf{A}(\mathbf{B}\mathbf{A} + c\mathbf{I})^{-1}.$$

This is exactly equal to the ridge regression solution (3) with $\lambda = \frac{\sigma^2}{s^2}$! Why is this?

Consider finding the *maximum a posteriori* estimator of $\mathbf{w}$ using the simple prior (4). Bayes' theorem tells us that the posterior distribution $p(\mathbf{w} \mid \mathcal{D})$ is proportional to the likelihood times the prior:

$$p(\mathbf{w} \mid \mathbf{X}, \mathbf{y}, \sigma^2, s^2) \propto p(\mathbf{y} \mid \mathbf{X}, \mathbf{w}, \sigma^2) p(\mathbf{w} \mid s^2).$$

The distribution of $\mathbf{y}$ given $\mathbf{w}$ is simple:

$$p(\mathbf{y} \mid \mathbf{X}, \mathbf{w}, \sigma^2) = \mathcal{N}(\mathbf{y}; \mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}) = \prod_{i=1}^{N} \mathcal{N}(y_i; \mathbf{x}_i^\top\mathbf{w}, \sigma^2).$$

Rather than maximizing the posterior directly, we may instead maximize its logarithm:

$$\begin{aligned}\hat{\mathbf{w}}_{\text{MAP}} &= \arg\max_{\mathbf{w}} -\frac{1}{2\sigma^2}\sum_{i=1}^{N}(\mathbf{x}_i^\top \mathbf{w} - y_i)^2 - \frac{1}{2s^2}\sum_{i=1}^{d} w_i^2 \\ &= \arg\min_{\mathbf{w}} \sum_{i=1}^{N}(\mathbf{x}_i^\top \mathbf{w} - y_i)^2 + \frac{\sigma^2}{s^2}\|\mathbf{w}\|_2^2,\end{aligned}$$

where we multiplied by $-2\sigma^2$ to derive the second line. This is exactly the ridge regression objective. Therefore ridge regression with ℓ^2 regularization can be seen from the Bayesian perspective as placing a Gaussian prior on $\mathbf{w}$ and finding the MAP estimator. In general, many regularization methods can be interpreted as implicitly choosing a particular likelihood for the data and prior for the parameters of a model and maximizing the posterior density. For example, ℓ^1 regularization (as seen in LASSO) is equivalent to placing a Laplace distribution prior on the weight parameters, encouraging a sparse solution.

Continuing with this argument, we can interpret the squared loss function $\sum_{i=1}^{N}(\mathbf{x}^\top \mathbf{w} - y_i)^2$ as coming from an iid Gaussian noise assumption. In general, loss functions in minimization problems such as (3) can often be interpreted as negative log likelihoods arising from an implicit independent noise assumption.