Regression review

Recall that regression is a form of supervised learning. Given some data x (typically a scalar or a vector), we're trying to predict a single value y. You've seen cases where y is a real number (linear regression) or a binary value $\in \{0, 1\}$ (logistic regression). Let's briefly review the setup for linear regression.

We have a collection of data $(x_1, y_1), \ldots, (x_n, y_n)$. We're trying to predict y_i from x_i, but our prediction won't be perfect. We'll use the notation $\hat{y}_i$ to represent the predicted value for data point i. We start by discussing how to get the predictions, and then move on to the relationship between the predictions and the actual observed values.

One dimension

In one dimension, we have data in the form $(x_1, y_1), \ldots, (x_n, y_n)$, where each x_i is a scalar and each y_i is a scalar. We form a linear prediction

$$\hat{y}_i = ax_i + b,$$

where a is a slope and b is an intercept: one-dimensional linear regression involves computing these.

Multiple dimensions

In multiple linear regression, we still have data in the form $(x_1, y_1), \ldots, (x_n, y_n)$, but now each $x_i \in \mathbb{R}^d$ is a d-dimensional vector. We can write

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$$

Each entry of this vector corresponds to a different feature that we're using in our prediction. See the examples section below for some concrete examples of what this might look like.

In multiple linear regression, we form our prediction for data point i, y_i, as follows:

$$\hat{y}_i = \sum_j \beta_j x_{ij}$$

The d-dimensional vector $\beta = (\beta_1, \ldots, \beta_d)$ contains the coefficients for each feature: linear regression involves figuring out what these are. We can write this in vector notation using the vectors β and x_i:

$$\hat{y}_i = \beta^T x_i = x_i^T \beta$$

We can take this notation one step further, and construct a matrix with all the x values for all data points and all features.

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}$$

One entry of this matrix, x_{ij}, represents feature j for data point i. If we consider the entire vector of predictions $\hat{y} = (\hat{y}_1, \ldots, \hat{y}_n)$, we can write the predictions in a fully vectorized way:

$$\hat{y} = X\beta$$

Likelihoods and loss functions

In order to compute the vector of coefficients β, we need some way to connect the predictions $\hat{y}_i$ (which are based on β) with the actual observed values y_i. You've seen two ways of doing this:

1. A loss function between the prediction $\hat{y}$ and the observed value y; we can minimize this loss function to find β.
2. A probabilistic model that describes the errors $\epsilon = y - \hat{y}$ as random variables, and tries to maximize the likelihood of the data under that model.

Loss functions

Recall that in linear regression, we try to find the value of y that minimizes the mean squared error (MSE). We can write the RMSE as follows:

$$\text{MSE} = \frac{1}{n}\sum_i (y_i - \beta^T x_i)^2$$

We can also write it as the ℓ_2 norm of the vector $y - \hat{y}$:

$$\text{MSE} = \frac{1}{n}\|y - \hat{y}\|_2^2 = \frac{1}{n}\|y - X\beta\|_2^2$$

where for any vector z, the ℓ_2 norm of z is $\|z\|_2 = \sqrt{\sum_i z_i^2}$.

We want to choose a value for β that makes this as small as possible:

$$\hat{\beta} = \text{argmin}_\beta \|y - X\beta\|_2^2$$

Likelihood and noise

We can also describe the errors in our model:

$$y_i = \beta^T x_i + \epsilon_i,$$

where $\epsilon_i \sim N(0, \sigma^2)$ is a random variable that represents the noise, or error, in the observed value. We can vectorize this noise, too: we'll write $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$ so that the vector ϵ has a multivariate normal distribution $\epsilon \sim N(0, \sigma^2 I_n)$. We can then write:

$$y = X\beta + \epsilon,$$

or equivalently, using properties of the normal distribution,

$$y|\beta \sim N(X\beta, \sigma^2 I_n).$$

Under this model, one reasonable way to estimate β is to choose the value that maximizes the likelihood. When choosing a value of β to maximize the likelihood, we note that we don't actually care about the normalizing constant in the normal distribution. So, we can write:

$$\hat{\beta} = \text{argmax}_\beta \exp\left\{-\frac{1}{2}(y - X\beta)^T(\sigma^2 I_n)^{-1}(y - X\beta)^T\right\} \tag{1}$$

$$= \text{argmax}_\beta \exp\left\{-\frac{1}{2\sigma^2}\|y - X\beta\|_2^2\right\} \tag{2}$$

Maximizing a quantity inside an exponential is hard. So, we'll take advantage of the fact that the $\log$ function is monotonically increasing. Furthermore, we'll make this a minimization rather than a maximization. In general, for any well-behaved function f:

$$\text{argmax}_\theta f(\theta) = \text{argmax}_\theta \log(f(\theta)) \tag{3}$$

$$= \text{argmin}_\theta \left[-\log(f(\theta))\right] \tag{4}$$

So, we can write:

$$\hat{\beta} = \text{argmax}_\beta \exp\left\{-\frac{1}{2\sigma^2}\|y - X\beta\|_2^2\right\} \tag{5}$$

$$= \text{argmin}_\beta \left[\frac{1}{2\sigma^2}\|y - X\beta\|_2^2\right] \tag{6}$$

$$= \text{argmin}_\beta \|y - X\beta\|_2^2 \tag{7}$$

So, we've found that maximizing the Gaussian likelihood of the data is exactly equivalent to minimizing the squared loss. This is true in general for regression problems: we can arrive at the same answer by either choosing a loss function and minimizing it, or choosing a corresponding likelihood and maximizing it.

Logistic regression

Recall that in logistic regression, we're trying to predict binary outputs: $y_i \in \{0, 1\}$. We're trying to predict the **probability** that y_i will be 1, which we'll call $\hat{y}$:

$$\hat{y}_i = \sigma(\beta^T x_i),$$

where σ is the sigmoid function, which converts real values to values between 0 and 1. To find β, we minimize the binary cross-entropy loss:

$$\hat{\beta} = \text{argmin}_\beta \sum_i -\left[y_i \ln(\hat{y}_i) + (1 - y_i)\ln(1 - \hat{y}_i)\right]$$

You'll show on the discussion worksheet that if we assume the likelihood model for y is Bernoulli with parameter $\sigma(\beta^T x_i)$, then maximizing the likelihood is equivalent to minimizing the binary cross-entropy loss.

For a deeper refresher on logistic regression, see Chapter 23 of the Data 100 textbook. Note that our notation is slightly different:

- We're using β instead of θ for the coefficients
- We're using $\hat{y}$ for the predictions instead of $f_{\hat{\theta}}$.