
Generalized Linear Regression with Regularization

Zoya Byliskii

March 3, 2015

1 BASIC REGRESSION PROBLEM

Note: In the following notes I will make explicit what is a vector and what is a scalar using vector notation, to avoid confusion between variables. I will occasionally expand out the vector notation to make the linear algebra operations more explicit.

You are given n data points: d -dimensional feature vectors and corresponding real-valued labels $\{\bar{x}^{(t)}, y^{(t)}\}$, $t = \{1, \dots, n\}$. Your goal is to find a linear combination of the features (feature vectors elements/coordinates) that is best able to predict each label. In other words, you want to discover the parameter vector $\bar{\theta}$ that allows you to make the most accurate predictions:

$$\bar{x}^{(t)} \cdot \bar{\theta} \approx y^{(t)}, \forall t = \{1, \dots, n\} \quad (1.1)$$

Note: we are ignoring offset for now, and solving a problem without constant offset θ_0 .

In other words, we want to approximate the labels we have for our n data points by finding the best-fitting $\bar{\theta}$. Expanding out the vector notation in eq. 1.1, we want:

$$\begin{aligned} x_1^{(1)}\theta_1 + x_2^{(1)}\theta_2 + \dots + x_d^{(1)}\theta_d &\approx y^{(1)} \\ x_1^{(2)}\theta_1 + x_2^{(2)}\theta_2 + \dots + x_d^{(2)}\theta_d &\approx y^{(2)} \\ &\vdots \\ x_1^{(n)}\theta_1 + x_2^{(n)}\theta_2 + \dots + x_d^{(n)}\theta_d &\approx y^{(n)} \end{aligned}$$

This is just a linear system of n equations in d unknowns. So, we can write this in matrix form:

$$\begin{pmatrix} \overline{x}^{(1)} \\ \overline{x}^{(2)} \\ \vdots \\ \overline{x}^{(n)} \end{pmatrix} \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_d \end{pmatrix} \approx \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix} \quad (1.2)$$

Or more simply as:

$$X\overline{\theta} \approx \overline{y} \quad (1.3)$$

Where X is our data matrix.

Note: the horizontal lines in the matrix help make explicit which way the vectors are stacked in the matrix. In this case, our feature vectors $\overline{x}^{(t)}$ make up the rows of our matrix, and the individual features/coordinates are the columns.

Consider the dimensions of our system:

$$\begin{array}{|c|} \hline n \times d \\ \hline \end{array} \begin{array}{|c|} \hline d \times 1 \\ \hline \end{array} = \begin{array}{|c|} \hline n \times 1 \\ \hline \end{array}$$

What happens if $n = d$?

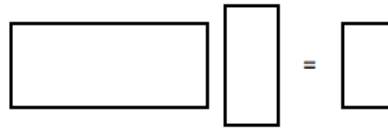
$$\begin{array}{|c|} \hline \\ \hline \end{array} \begin{array}{|c|} \hline \\ \hline \end{array} = \begin{array}{|c|} \hline \\ \hline \end{array}$$

X is square matrix, and as long as all the data points (rows) of X are linearly independent, then X is invertible and we can solve for $\overline{\theta}$ exactly:

$$\overline{\theta} = X^{-1}\overline{y} \quad (1.4)$$

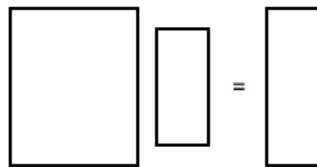
Of course a $\overline{\theta}$ that exactly fits all the training data is not likely to generalize to a test set (a novel set of data: new feature vectors $\overline{x}^{(t)}$). Thus we might want to impose some regularization to avoid overfitting (will be discussed later in this document), or to subsample the data or perform some cross-validation (leave out part of the training data when solving for the parameter $\overline{\theta}$).

What happens if $n < d$?


$$\boxed{} \boxed{} = \boxed{}$$

In this case, we have fewer data points than feature dimensions - less equations than unknowns. Think: how many degree-3 polynomials can pass through 2 points? An infinite number. Similarly, in this case, we can have an infinite number of solutions. Our problem is undefined in this case. What can we do? Well, we can collect more data, we can replicate our existing data while adding some noise to it, or we can apply some methods of feature selection (a research area in machine learning) to select out the features (columns of our data matrix) that carry the most weight for our problem. We have to regularize even more severely in this case, because there's an infinite number of ways to overfit the training data.

What happens if $n > d$?


$$\boxed{} \boxed{} = \boxed{}$$

Now we have more data than features, so the more data we have, the less likely we are to overfit the training set. In this case, we will not be able to find an exact solution: a $\bar{\theta}$ that satisfies eq. 1.1. Instead, we will look for a $\bar{\theta}$ that is best in the least-squares sense. It turns out, as we will see later in this document, that this $\bar{\theta}$ can be obtained by solving the modified system of equations:

$$X^T X \bar{\theta} = X^T \bar{y} \tag{1.5}$$

This is called the normal equations. Notice that $X^T X$ is a square $d \times d$, invertible matrix¹.

We can now solve for $\bar{\theta}$ as follows:

$$\bar{\theta} = (X^T X)^{-1} X^T \bar{y} \tag{1.6}$$

Where $(X^T X)^{-1} X^T$ is often denoted as X^+ and is called the pseudoinverse of X .

¹An easy-to-follow proof is provided here: <https://www.khanacademy.org/math/linear-algebra/matrix-transformations/matrix-transpose/v/lin-alg-showing-that-a-transpose-x-a-is-invertible>

Let us now arrive at this solution by direct optimization of the least squares objective:

$$J(\bar{\theta}) = \frac{1}{n} \sum_{t=1}^n \frac{1}{2} \left[y^{(t)} - \bar{\theta} \cdot \bar{x}^{(t)} \right]^2 \quad (1.7)$$

Notice that this is just a way of summarizing the constraints in eq. 1.1. We want $\bar{\theta} \cdot \bar{x}^{(t)}$ to come as close as possible to $y^{(t)}$ for all t , so we minimize the sum of squared errors.

Note: the $\frac{1}{2}$ term is only for notational convenience (for taking derivatives). It is just a constant scaling factor of the final $\bar{\theta}$ we obtain. Also, the $\frac{1}{n}$ term is not important for this form of regression without regularization, as it will cancel out. In the form with regularization, it just rescales the regularization parameter by the amount of data points n (we'll see this later in this document). In any case, you might see formulations of regression with or without this term, but this will not make a big difference to the general form of the problem.

We want to minimize $J(\bar{\theta})$, and so we set the gradient of this function to zero: $\nabla_{\theta}(J(\bar{\theta})) = 0$. This is equivalent to solving the following system of linear equations:

$$\begin{pmatrix} \frac{\partial}{\partial \theta_1} (J(\bar{\theta})) \\ \frac{\partial}{\partial \theta_2} (J(\bar{\theta})) \\ \vdots \\ \frac{\partial}{\partial \theta_d} (J(\bar{\theta})) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad (1.8)$$

Let's consider a single one of these equations: i.e. solve for a particular partial $\frac{\partial}{\partial \theta_i} (J(\bar{\theta}))$. For clarity, consider expanding the vector notation in 1.7:

$$J(\bar{\theta}) = \frac{1}{n} \sum_{t=1}^n \frac{1}{2} \left[y^{(t)} - \left(x_1^{(t)} \theta_1 + x_2^{(t)} \theta_2 + \dots + x_d^{(t)} \theta_d \right) \right]^2 \quad (1.9)$$

Thus, by the chain rule:

$$\frac{\partial}{\partial \theta_i} (J(\bar{\theta})) = \frac{1}{n} \sum_{t=1}^n \left[y^{(t)} - \left(x_1^{(t)} \theta_1 + x_2^{(t)} \theta_2 + \dots + x_d^{(t)} \theta_d \right) \right] \left(-x_i^{(t)} \right) \quad (1.10)$$

$$= \frac{1}{n} \sum_{t=1}^n \left[y^{(t)} - \left(\bar{x}^{(t)} \right)^T \bar{\theta} \right] \left(-x_i^{(t)} \right) \quad (1.11)$$

Note: we just rewrote the dot product $\bar{\theta} \cdot \bar{x}^{(t)}$ in equivalent matrix form: $\left(\bar{x}^{(t)} \right)^T \bar{\theta}$ because we will be putting our whole system of equations in matrix form in the following calculations.

Since we want to set all the partials to 0, it follows that:

$$\frac{1}{n} \sum_{t=1}^n \left[y^{(t)} - \left(\bar{x}^{(t)} \right)^T \bar{\theta} \right] \left(-x_i^{(t)} \right) = 0 \quad (1.12)$$

$$\frac{1}{n} \sum_{t=1}^n x_i^{(t)} \left(\bar{x}^{(t)} \right)^T \bar{\theta} = \frac{1}{n} \sum_{t=1}^n x_i^{(t)} y^{(t)} \quad (1.13)$$

So we can rewrite the system in eq. 1.14 as:

$$\begin{pmatrix} \frac{1}{n} \sum_{t=1}^n x_1^{(t)} (\bar{x}^{(t)})^T \bar{\theta} \\ \frac{1}{n} \sum_{t=1}^n x_2^{(t)} (\bar{x}^{(t)})^T \bar{\theta} \\ \vdots \\ \frac{1}{n} \sum_{t=1}^n x_d^{(t)} (\bar{x}^{(t)})^T \bar{\theta} \end{pmatrix} = \begin{pmatrix} \frac{1}{n} \sum_{t=1}^n x_1^{(t)} y^{(t)} \\ \frac{1}{n} \sum_{t=1}^n x_2^{(t)} y^{(t)} \\ \vdots \\ \frac{1}{n} \sum_{t=1}^n x_d^{(t)} y^{(t)} \end{pmatrix} \quad (1.14)$$

Which is equivalent to the following condensed form:

$$\frac{1}{n} \sum_{t=1}^n \bar{x}^{(t)} (\bar{x}^{(t)})^T \bar{\theta} = \frac{1}{n} \sum_{t=1}^n \bar{x}^{(t)} y^{(t)} \quad (1.15)$$

Dropping the $\frac{1}{n}$ term:

$$\sum_{t=1}^n \bar{x}^{(t)} (\bar{x}^{(t)})^T \bar{\theta} = \sum_{t=1}^n \bar{x}^{(t)} y^{(t)} \quad (1.16)$$

In other words, this equation is what we obtain by setting $\nabla_{\theta}(J(\bar{\theta})) = 0$.

Let us write this equation out explicitly to see how we can rewrite it further in matrix form. First, consider writing out the left-hand-side of eq. 1.16:

$$\begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_d^{(1)} \end{pmatrix} (x_1^{(1)} x_2^{(1)} \dots x_d^{(1)}) \bar{\theta} + \begin{pmatrix} x_1^{(2)} \\ x_2^{(2)} \\ \vdots \\ x_d^{(2)} \end{pmatrix} (x_1^{(2)} x_2^{(2)} \dots x_d^{(2)}) \bar{\theta} + \dots + \begin{pmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_d^{(n)} \end{pmatrix} (x_1^{(n)} x_2^{(n)} \dots x_d^{(n)}) \bar{\theta} \quad (1.17)$$

Convince yourself that this is just:

$$\begin{pmatrix} | & | & \dots & | \\ \bar{x}^{(1)} & \bar{x}^{(2)} & \dots & \bar{x}^{(n)} \\ | & | & \dots & | \end{pmatrix} \begin{pmatrix} \text{---} \bar{x}^{(1)} \text{---} \\ \text{---} \bar{x}^{(2)} \text{---} \\ \vdots \\ \text{---} \bar{x}^{(n)} \text{---} \end{pmatrix} \bar{\theta} \quad (1.18)$$

These matrices should be familiar from before. We can rewrite eq. 1.18 as $X^T X \bar{\theta}$.

Next, consider writing out the right-hand-side of eq. 1.16:

$$\begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_d^{(1)} \end{pmatrix} y^{(1)} + \begin{pmatrix} x_1^{(2)} \\ x_2^{(2)} \\ \vdots \\ x_d^{(2)} \end{pmatrix} y^{(2)} + \dots + \begin{pmatrix} x_1^{(n)} \\ x_2^{(n)} \\ \vdots \\ x_d^{(n)} \end{pmatrix} y^{(n)} \quad (1.19)$$

Convince yourself that this is just:

$$\begin{pmatrix} \left| \begin{array}{c} \bar{x}^{(1)} \\ | \\ | \end{array} \right| & \left| \begin{array}{c} \bar{x}^{(2)} \\ | \\ | \end{array} \right| & \dots & \left| \begin{array}{c} \bar{x}^{(n)} \\ | \\ | \end{array} \right| \end{pmatrix} \bar{y} \quad (1.20)$$

Which is nothing more than $X^T \bar{y}$.

Putting together eq. 1.18 and eq. 1.20, we get exactly eq. 1.5, and so:

$$\bar{\theta} = (X^T X)^{-1} X^T \bar{y}$$

as the least-squares solution of our regression problem (no offset, no regularization).

2 REGRESSION WITH REGULARIZATION

If we overfit our training data, our predictions may not generalize very well to novel test data. For instance, if our training data is somewhat noisy (real measurements are always somewhat noisy!), we don't want to fit our training data perfectly - otherwise we might generate very bad predictions for real signal. It is worse to overshoot than undershoot predictions when errors are measured by squared error. Being more guarded in how you predict, i.e., keeping $\bar{\theta}$ small, helps reduce generalization error. We can enforce the constraint to keep $\bar{\theta}$ small by adding a regularization term to 1.7:

$$J(\bar{\theta}) = \frac{1}{n} \sum_{t=1}^n \frac{1}{2} \left[y^{(t)} - \bar{\theta} \cdot \bar{x}^{(t)} \right]^2 + \frac{\lambda}{2} \|\bar{\theta}\|^2 \quad (2.1)$$

Since $\frac{\lambda}{2} \|\bar{\theta}\|^2 = \frac{\lambda}{2} \theta_1^2 + \frac{\lambda}{2} \theta_2^2 + \dots + \frac{\lambda}{2} \theta_d^2$ this contributes a single additional term to eq. 1.11:

$$\frac{\partial}{\partial \theta_i} (J(\bar{\theta})) = \frac{1}{n} \sum_{t=1}^n \left[y^{(t)} - (\bar{x}^{(t)})^T \bar{\theta} \right] \left(-x_i^{(t)} \right) + \lambda \theta_i \quad (2.2)$$

So similarly to eq. 1.13, setting the partial to zero:

$$\frac{1}{n} \sum_{t=1}^n x_i^{(t)} \left(\bar{x}^{(t)} \right)^T \bar{\theta} + \lambda \theta_i = \frac{1}{n} \sum_{t=1}^n x_i^{(t)} y^{(t)} \quad (2.3)$$

Thus, collapsing the equations for all partials:

$$\frac{1}{n} \sum_{t=1}^n \bar{x}^{(t)} \left(\bar{x}^{(t)} \right)^T \bar{\theta} + \lambda \bar{\theta} = \frac{1}{n} \sum_{t=1}^n \bar{x}^{(t)} y^{(t)} \quad (2.4)$$

And in matrix form:

$$\frac{1}{n} X^T X \bar{\theta} + \lambda \bar{\theta} = \frac{1}{n} X^T \bar{y} \quad (2.5)$$

$$\left[\frac{1}{n} X^T X + \lambda I \right] \bar{\theta} = \frac{1}{n} X^T \bar{y} \quad (2.6)$$

$$(2.7)$$

Which gives us the solution of our least-squares regression problem with regularization:

$$\bar{\theta} = \left[\frac{1}{n} X^T X + \lambda I \right]^{-1} \frac{1}{n} X^T \bar{y}$$

Note: You can see that the $\frac{1}{n}$ term was not dropped in the regularized form of the problem, but is equivalent to a rescaling of λ .

Aside: how do we know $\frac{1}{n} X^T X + \lambda I$ is invertible? Here is a rough proof outline:

- First note that $X^T X$ is positive semidefinite
proof: $u^T X^T X u = \|Xu\|^2 \geq 0$ so all eigenvalues must be ≥ 0
- The eigenvalues of $X^T X + \lambda I$ are $\mu_i + \lambda$, where μ_i are eigenvalues of $X^T X$
proof: $X^T X u = \mu u$ implies $(X^T X + \lambda I) u = (\mu + \lambda) u$
- All eigenvalues of $X^T X$ are strictly positive, so it must be invertible
proof: $\lambda > 0$, and so $\mu_i + \lambda > 0$ for every i

3 REGRESSION WITH REGULARIZATION AND OFFSET

The offset parameter can be thought of as adjusting to the magnitude of the data. It is a single scalar, and we do not want to penalize its size during regularization, because a large offset might allow many of the other parameters to be much smaller and still provide a good fit for the data. The calculation for the least-squares solution with regularization and offset is similar to the calculation without offset.

$$J(\bar{\theta}) = \frac{1}{n} \sum_{t=1}^n \frac{1}{2} \left[y^{(t)} - (\bar{\theta} \cdot \bar{x}^{(t)} + \theta_o) \right]^2 + \frac{\lambda}{2} \|\bar{\theta}\|^2 \quad (3.1)$$

Setting $\nabla_{\theta}(J(\bar{\theta})) = 0$ as before, we have:

$$\frac{1}{n} \sum_{t=1}^n \bar{x}^{(t)} (\bar{x}^{(t)})^T \bar{\theta} + \frac{1}{n} \theta_o \sum_{t=1}^n \bar{x}^{(t)} + \lambda \bar{\theta} = \frac{1}{n} \sum_{t=1}^n \bar{x}^{(t)} y^{(t)} \quad (3.2)$$

And in matrix form:

$$\frac{1}{n} X^T X \bar{\theta} + \frac{1}{n} \theta_o X^T \bar{1} + \lambda \bar{\theta} = \frac{1}{n} X^T \bar{y} \quad (3.3)$$

$$\left(\frac{1}{n} X^T X + \lambda \right) \bar{\theta} + \left(\frac{1}{n} X^T \bar{1} \right) \theta_o = \frac{1}{n} X^T \bar{y} \quad (3.4)$$

Where $\bar{1}$ is the vector composed of all ones (here with dimension $n \times 1$), and so $X^T \bar{1} = \sum_{t=1}^n \bar{x}^{(t)}$.

We separately minimize $J(\bar{\theta})$ with respect to θ_o (which does not appear in the regularizer):

$$\frac{\partial}{\partial \theta_o} (J(\bar{\theta})) = \frac{1}{n} \sum_{t=1}^n \left[y^{(t)} - (\bar{x}^{(t)})^T \bar{\theta} - \theta_o \right] (-1) \quad (3.5)$$

Setting this partial to 0 we get:

$$\theta_o = \frac{1}{n} \sum_{t=1}^n \left[y^{(t)} - (\bar{x}^{(t)})^T \bar{\theta} \right] \quad (3.6)$$

$$= \frac{1}{n} \bar{y}^T \bar{1} - \frac{1}{n} (X \bar{\theta})^T \bar{1} \quad (3.7)$$

Note that we now have 2 equations (3.4 and 3.7) in 2 unknowns: θ and θ_o that can be solved algebraically.

4 ADDITIONAL RESOURCES

- Matrix cookbook: <http://www.mit.edu/~wingated/stuff_i_use/matrix_cookbook.pdf>.
- Matrix approach to linear regression with a statistical perspective: <http://www.maths.qmul.ac.uk/~bb/SM_I_2013_LecturesWeek_6.pdf>.