Linear Regression: Matrix Method

by Dan Wilhelm

`Here, we show the matrix equations for doing linear regression, how to interpret each matrix, and we provide a simple example which can be done by hand.`

In linear regression, we attempt to find a linear equation that best "fits" our data points. In other words, we are trying to estimate the data via an estimator function $\hat{y}(x_1, x_2, \ldots, x_n)$, where:

$$\hat{y}(x_1, x_2, \ldots, x_n) = \beta_0 + x_1\beta_1 + x_2\beta_2 + \ldots + x_n\beta_n$$

Note that each feature x_i has an associated β_i. A large β_i means that the x_i directly affects the estimated target $\hat{y}$ by a large positive amount. By comparing the magnitudes of each β_i, we can tell how influential each feature x_i is to the final result.

For example, suppose y is salary and x_1 is a binary value, where 0 is male and 1 is female. If β_1 is 200, then according to this model being female predicts a higher salary by $200. Note that the β-values in a linear regression model are hence easy to interpret.

Matrix Interpretation

Suppose we have a single initial data point, with n features and one target y: $(x_1, x_2, \ldots, x_n, y)$. Then, we can store these immediately as column vectors (where each data point is one row), as follows:

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} y \end{bmatrix}$$

We are trying to solve for one β_i for each feature x_i. In other words, we are trying to solve for the column vector $\mathbf{B}$:

$$\mathbf{B} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \ldots \\ \beta_n \end{bmatrix}$$

However, note that the $\mathbf{B}$ matrix is $(n + 1)$ x 1, whereas $\mathbf{X}$ is only 1 x n. If we force x_0 to be 1, then the two will have matching dimensions for a matrix multiply. In the next section, we will multiply $\hat{\mathbf{Y}} = \mathbf{X_b}\mathbf{B}$, so it will result in $1\beta_0 + x_1\beta_1 + x_2\beta_2 + \ldots + x_n\beta_n$. This allows us to have a bias!

The Matrix Equations

We will call the new $\mathbf{X}$ with a 1-column appended to it, the column vector $\mathbf{X_b}$:

$$\mathbf{X_b} = \begin{bmatrix} 1 & x_1 & x_2 & \ldots & x_n \end{bmatrix}$$

So now, let's rewrite our linear estimator equation from above in matrix form:

$$\hat{\mathbf{Y}} = \mathbf{X_b}\mathbf{B}$$

$$\begin{bmatrix} \hat{y} \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_2 & \ldots & x_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \ldots \\ \beta_n \end{bmatrix}$$

i.e.

$$\hat{y}(x_1, x_2, \ldots, x_n) = \beta_0 + x_1\beta_1 + x_2\beta_2 + \ldots + x_n\beta_n$$

Now, we can compute the $\mathbf{B}$ using the following equation. In fact, the secret to linear regression is that each β_i can be solved for exactly, without multiple iterations, as required by most other learning methods. This is possible because the system is linear.

$$\mathbf{B} = (\mathbf{X_b'}\mathbf{X})^{-1}\mathbf{X'}\mathbf{Y}$$

See [http://isites.harvard.edu/fs/docs/icb.topic515975.files/OLSDerivation.pdf] for a derivation.

Example

Let's make some really simple sample data and compute these equations by hand so we better understand the matrix equations.

We will make up two points that fall on the line $y = 2x + 8$. Then, if we use linear regression on those two data points, we should compute the β values 8 and 2!

Let's get two data points that fall on the line – $(-4, 0)$ and $(2, 12)$. In this example, note that we have a single feature x_1 and a single target y. So:

$$\mathbf{X} = \begin{bmatrix} -4 \\ 2 \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} 0 \\ 12 \end{bmatrix}$$

For $\mathbf{X}$, the columns are the features, and the rows are the datapoints! Now let's add the 1-column for the biases:

$$\mathbf{X_b} = \begin{bmatrix} 1 & -4 \\ 1 & 2 \end{bmatrix}$$

So:

$$\mathbf{X_b'} = \begin{bmatrix} 1 & 1 \\ -4 & 2 \end{bmatrix}$$

Let's check that what we've done so far is correct, by computing $\hat{\mathbf{Y}} = \mathbf{X_b}\mathbf{B}$. Note that here, we know that $\mathbf{B} = \begin{bmatrix} 8 \\ 2 \end{bmatrix}$, since we designed this data to correspond to the equation $y = 2x + 8$.

$$\hat{\mathbf{Y}} = \begin{bmatrix} 1 & -4 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 8 \\ 2 \end{bmatrix}$$

It works out as expected, yielding our original $\mathbf{Y} = \hat{\mathbf{Y}} = \begin{bmatrix} 0 \\ 12 \end{bmatrix}$. Now, let's verify that we can compute $\mathbf{B}$ using the formula $\mathbf{B} = (\mathbf{X_b'}\mathbf{X_b})^{-1}\mathbf{X_b'}\mathbf{Y}$:

$$\mathbf{X_b'}\mathbf{X_b} = \begin{bmatrix} 1 & 1 \\ -4 & 2 \end{bmatrix} \begin{bmatrix} 1 & -4 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 2 & -2 \\ -2 & 20 \end{bmatrix}$$

Note that you can doublecheck these using Wolfram Alpha at http://wolframalpha.com. An example of matrix input to Wolfram Alpha which multiplies the above matrices is:

`{{1,1}, {-4,2}} * {{1,-4}, {1,2}}`.

Inverting that (either by using the 2x2 inversion method or via Wolfram Alpha using ^–1) yields:

$$(\mathbf{X_b'}\mathbf{X_b})^{-1} = \frac{1}{18}\begin{bmatrix} 10 & 1 \\ 1 & 1 \end{bmatrix}$$

Then multiplying this by $\mathbf{X_b'}\mathbf{Y_b}$ to solve for $\mathbf{B}$:

$$\mathbf{B} = (\mathbf{X_b'}\mathbf{X_b})^{-1}\mathbf{X_b'}\mathbf{Y} = \frac{1}{18}\begin{bmatrix} 10 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ -4 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 12 \end{bmatrix} = \begin{bmatrix} 8 \\ 2 \end{bmatrix}$$

This yields $\mathbf{B} = \begin{bmatrix} 8 \\ 2 \end{bmatrix}$, as desired.

Now that we know the math works out, the orientation of our matrices, and how to insert the 1-column, we can write the Python code using numpy. (Left as an exercise.)