5.3 Geometry is not enough, Generalization is key

The geometry of linear regression is quite cool, but it also indicates a serious flaw if used uncritically. Suppose we add a new column to our measurement matrix B to produce a new measurement matrix B_+. This new column—our *fake feature*—could simply be a column of randomly generated numbers. Clearly every vector in $col(B)$ is in $col(B_+)$. Let the projection of the target $\mathbf{y}$ into $col(B)$ be $\mathbf{w}_{OLS} = B\mathbf{x}_{OLS}$.

Since $col(B) \subset col(B_+)$, $\mathbf{w}_{OLS} \in col(B_+)$ as well. So when we solve the regression with the new measurement matrix B_+, we wish to find

$$\min_{\mathbf{w} \in col(B_+)} ||Y - \mathbf{w}||^2,$$

but of course

$$\min_{\mathbf{w} \in col(B_+)} ||Y - \mathbf{w}||^2 \leq ||Y - \mathbf{w}_{OLS}||^2 = \min_{\mathbf{w} \in col(B)} ||Y - \mathbf{w}||^2,$$

since the left most side above minimizes over all points in $col(B_+)$ and $\mathbf{w}_{OLS}$ is one of the points in $col(B_+)$. The minimum cannot get worse than $\mathbf{w}_{OLS}$ at least.

So adding any feature, including randomly generating a fake one (as in the attached demo) to our training data, can only reduce the mean square error on training data. In the attached notebook, you will find that in the Boston housing dataset, we can drive down the mean square error on training examples to 0 by adding enough randomly generated features. This is clearly insane from the perspective of meaningful models.

So we have to look at the least squares approach more critically. The principle we used to choose the model $\mathbf{x}$ was Maximum Likelihood, but what we really want is that the model we choose is meaningful in some way. The most common way to phrase this is that the model $\mathbf{x}$ we pick must *generalize.* This means that if we choose $\mathbf{x}$ based on measurements B and observations Y and were subsequently given a new *test* measurement

$$\begin{bmatrix} b_1 & , \ldots, & b_k \end{bmatrix},$$

our prediction

$$\begin{bmatrix} b_1 & , \ldots, & b_k \end{bmatrix} \mathbf{x}$$

must not deviate too far from the ground truth on the test example.

Armed with this insight, how should we build a model? If we have a bunch of measurements B, we use the simple expedient of partitioning the measurements into *training* measurements B_{train} and *validation* B_v measurements, *i.e.,* we partition the rows of B into two disjoint sets. The targets for measurements B_{train} are in the vector Y_{train}, and for measurements B_v in Y_v. We choose the model $\mathbf{x}$ based on only B_{train} and Y_{train}, and use the model so obtained to predict the target on the validation set $B_v\mathbf{x}_{train}$. The training error

$$||Y_{train} - B_{train}\mathbf{x}_{train}||^2$$

measures the fit on the training data, but this is just geometry and the numerical magnitude does not have much significance. But the validation error,

$$||Y_v - B_v\mathbf{x}_{train}||^2$$

is interesting. Ideally, we would like to see the validation and training errors match up. If one is too far from the other, it is a sign that the geometry of linear regression is getting fooled, possibly from irrelevant or misleading features.