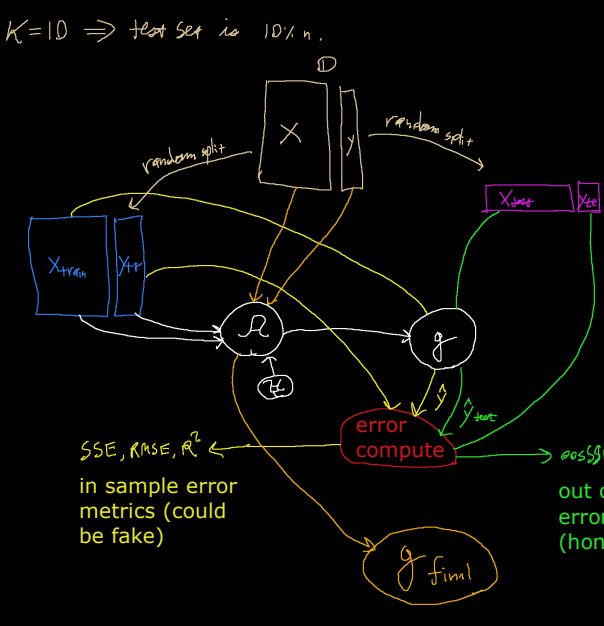
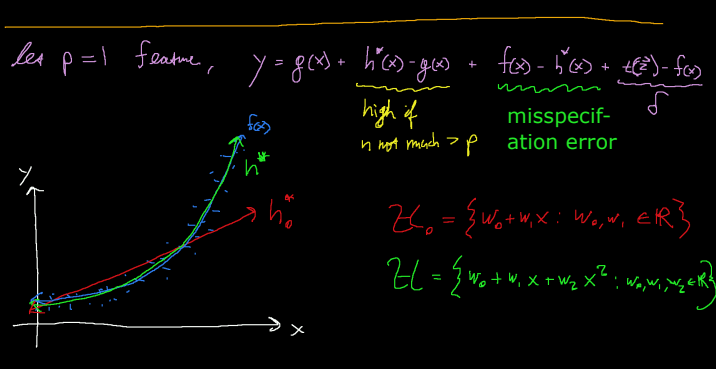


The g_final is the function used for future prediction. Its performance is at least as good as the oos metrics since you're running the same model fitting procedure but now n is slightly higher.



f(x) is not linear and therefore even the best possible linear model (h_0^*) will perform poorly. So why not allow for a more expressive candidate set? We can do that by expanding the basis / complexity in curlyH. For example, we now allow for a quadratic term so we can fit parabolic-shaped curves. This allows us to get closer to the real f (which may be very complex and nonlinear), reducing misspecification error. We now have p = 2 which is greater than p_raw = 1. We call this a "derived feature" in contrast to a "raw feature" (original). E.g. x_2 = g(x_1) = x_1^2. It's a transformation of a raw feature.

You're at liberty to use any transformed features you want. If they're useless, they appear as random noise and you overfit.

Using squares and cubes is a well-known modeling procedure called "polynomial regression".

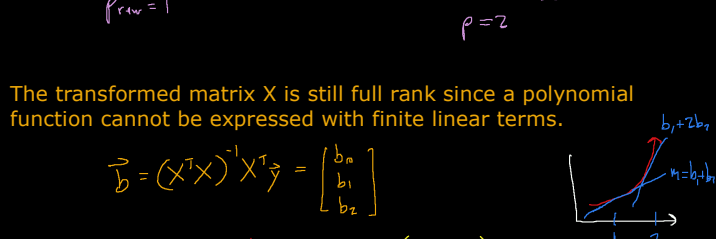
Is polynomial regression "linear"? Yes and no. "Yes" in the sense that you create a design matrix and use OLS and thus linear in the transformed features but "no" because the g model is not linear in the raw features.

Advanced math note: polynomial regression is a principled approach because of the Weierstrauss Approximation Thm (1885) which says that any continuous function f who domain is x in [a, b] can be approximated by a polynomial function p_d with arbitrary precision by picking d, its degree:

$$\forall \varepsilon > 0 \quad \forall x \in [a,b] \quad \exists d \;\Big|\; f(x) - p_d(x) \Big| < \varepsilon.$$

The Stone-Weierstrauss Thm (1937) generalizes the above. One implication of this thm is that a multivariate polynomial function can approximate any continuous function f(x_1, ..., x_p).

How do we do a polynomial regression of degree d. E.g. d = 2.



The transformed matrix X is still full rank since a polynomial function cannot be expressed with finite linear terms.

$$\vec{b} = (X^T X)^{-1} X^T \vec{y} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix}$$

$$g(x) = \hat{y} = b_0 + b_1 x + b_2 x^2 = b_0 + (b_1 + b_2 x)\, x$$

Can you do polynomial regression of degree d = 3? Yes. Same way! Just make a new feature and cube x_1. How far can you go in OLS? p = n-1 i.e. d = n-1. That would yield a perfect fit. Any higher d, and you can't invert X^T X. E.g. n = 5

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 & x_1^4 \\ 1 & x_2 & \cdot & \cdot & \cdot \\ 1 & x_3 & \cdot & \cdot & \cdot \\ 1 & x_4 & \cdot & \cdot & \cdot \\ 1 & x_5 & \cdot & \cdot & \cdot \end{bmatrix}$$

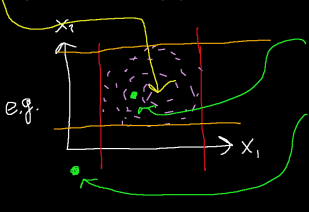
Is this full rank? This is a special matrix called a Vandermonde Matrix and it's proven to be full rank if:

$$\det[X] = \prod_{i=1}^{n} \prod_{j=1}^{n} x_j - x_i \neq 0.$$

Consider p raw features given by the columns of X. Define:

$$\text{Range}[X] = [x_{\cdot 1, \min}, x_{\cdot 1, \max}] \times [x_{\cdot 2, \min}, x_{\cdot 2, \max}] \times \cdots \times [x_{\cdot p, \min}, x_{\cdot p, \max}]$$

This is a hyperrectangle representing the space of x-vectors (observations) you've seen in your n examples.



"Interpolation" is when you predict for x-vectors inside the Range[X].

"Extrapolation" is when you predict for x-vectors outside the Range[X].

We build models to interpolate. Bad things could happen when you extrapolate. Different model fitting procedures (curlyA) extrapolate differently... beware!

We expanded the complexity of our candidate set curlyH using polynomials. But we found that high degree polynomials had unintended consequences (Runge's phenomenon). Is there another transformation of raw features that we can employ to expand curlyH? Of course... there are tons of functions! Exponentials, logs, sines, etc. Let's examine logs because they are very popular and very useful:

$$\ln(x+1) \approx x - \frac{x^2}{2} + \frac{x^3}{3} - \cdots \approx x \quad \text{if } x \approx 0$$

$$\Rightarrow \ln(x) = \ln((x+1) - 1) \approx x - 1 \quad \text{e.g. } \ln(1.02) = .019 \approx 1.02 - 1$$

consider the following linear model:

$$y = b_0 + b_1 \ln(x)$$

$$\Delta x = x_f - x_0 \overset{\text{e.g.}}{=} 1.02 - 1.00$$

% change in x

$$\Delta y = (b_0 + b_1 \ln(x_f)) - (b_0 + b_1 \ln(x_0)) = b_1 \ln\left(\frac{x_f}{x_0}\right) \approx b_1 \left(\frac{x_f}{x_0} - 1\right)$$

This simple log model can be approx interpreted as proportional change in x yields a change in y (in y's units) i.e. if x increases by 100%, y goes up by b_1.

Likewise you can do ln(y) = b_0 + b_1 x and this is approx interpreted as unit change in x yields b_1 proportion change in y and ln(y) = b_0 + b_1 ln(x) is approx interpreted as proportional change in x yields b_1 proportion change in y.