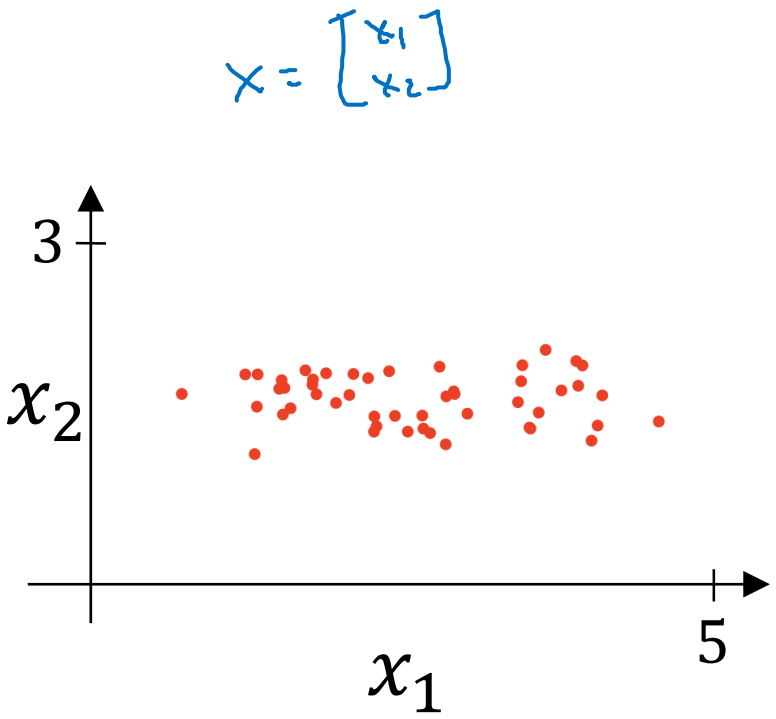
deeplearning.ai

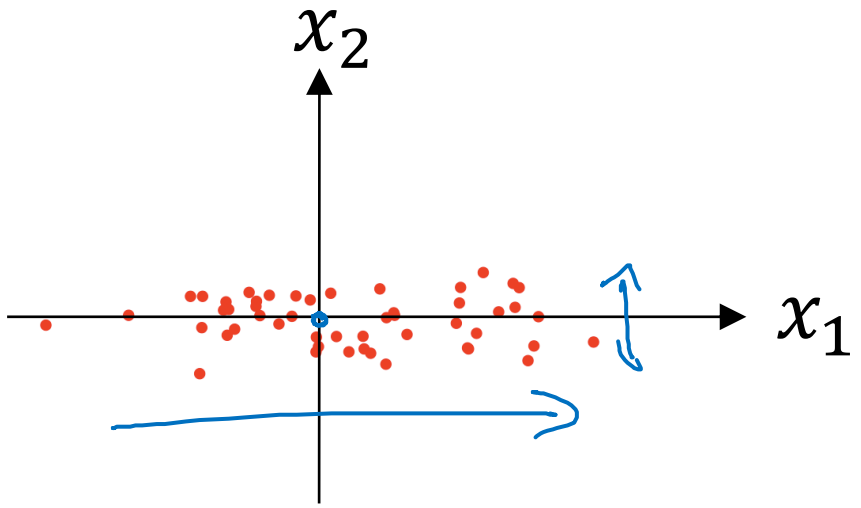
Setting up your optimization problem

Normalizing inputs

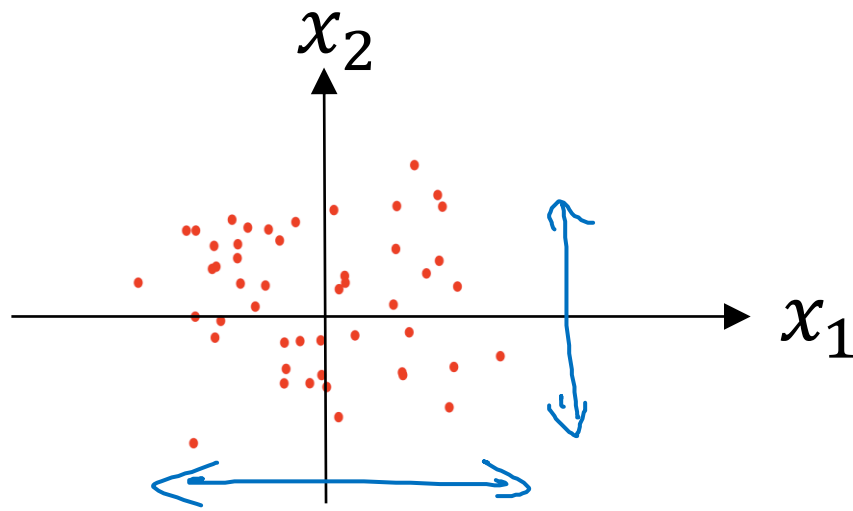
speed up your learning

Normalizing training sets





Now the variance of X1 and X2 are both equal to one.



You just move the training set until it has zero mean.

Subtract mean:

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$$

$$x := x - \mu$$

Normalize variance

$$\sigma^2 = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} **2$$

↖ element-wise

$$x /= \sigma^2$$

element wise Squaring
Notice we've already subtracted out the mean, so x(i) squared, element y squared is just the variances.

Because you want your data, both training and test examples ,to go through the same transformation defined by the same mu and sigma squared calculated on your training data.

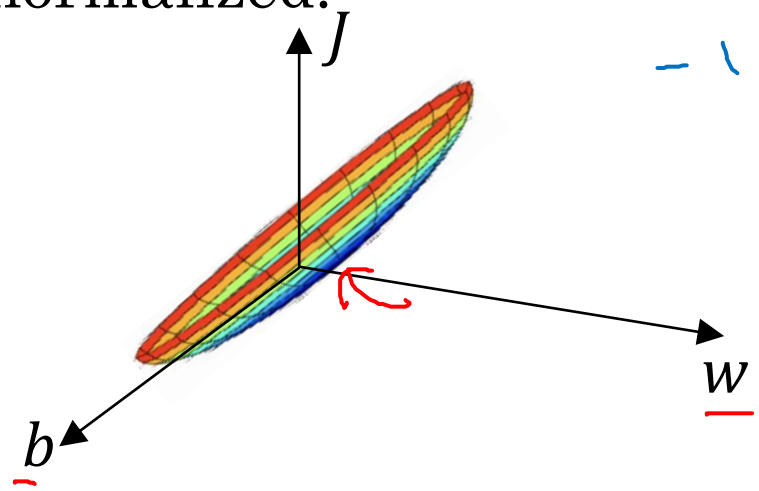
Use same μ, σ^2 to normalize test set.

Andrew Ng

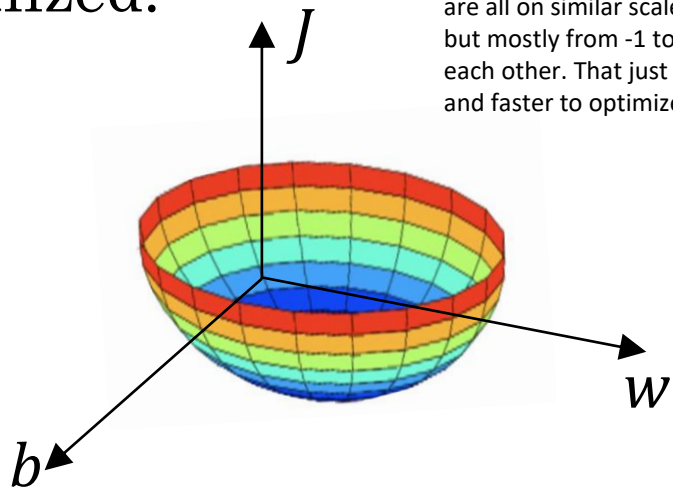
Why normalize inputs?

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}\left(\hat{y}^{(i)}, y^{(i)}\right)$$

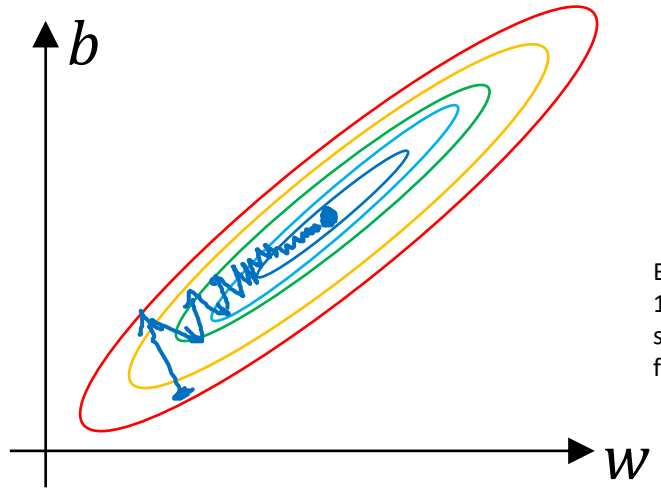
Unnormalized:



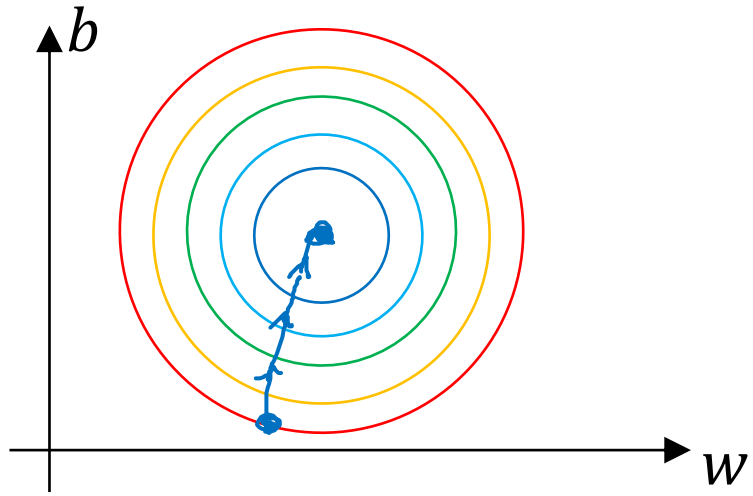
Normalized:



But the rough intuition that your cost function will be more round and easier to optimize when your features are all on similar scales, not from 1 to 1000, zero to one, but mostly from -1 to 1 or of about similar variance of each other. That just makes your cost function J easier and faster to optimize.



By just setting all of them to a 0 mean and say, variance 1, that just guarantees that all your features on a similar scale and will usually help your learning algorithm run faster.



Andrew Ng