LINEAR REGRESSION

11.4.2020

🇺🇸

RECAP

* receptive fields
  * what kind of stuff in the world does a neuron (or voxel, or whatever) respond to?
* spike-triggered average
  * limited applicability

RECAP: SPIKE-TRIGGERED AVERAGE (STA)

* e.g. imagine this is the data:

* $Y = [1,1,0,1,0]$
  $X_1 = [1,1,0,1,0]$
  $X_2 = [1,0,0,1,0]$

* STA would say that $beta_1 = 1$, and $beta_2 = 0.66$

* but a simpler explanation would be that $beta_2 = 0$, since X_1 already explains everything about Y

1D EXAMPLE

subject

stimulus

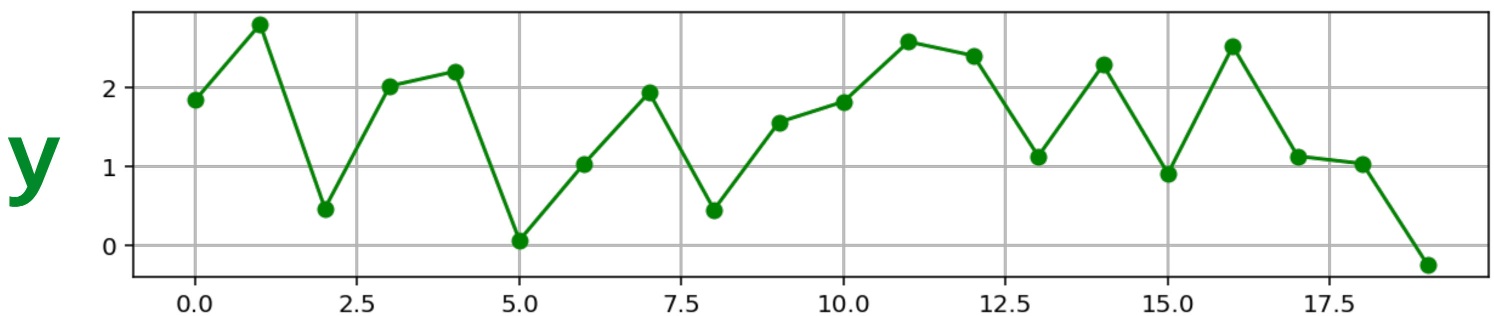
* **y** = output of a neuron that you are measuring

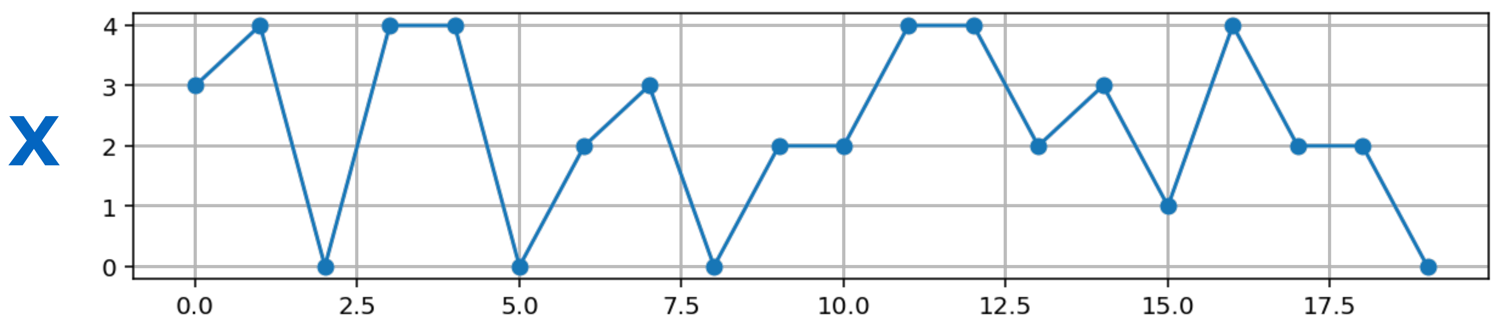
* **x** = how many times per second the screen flashes

1D EXAMPLE

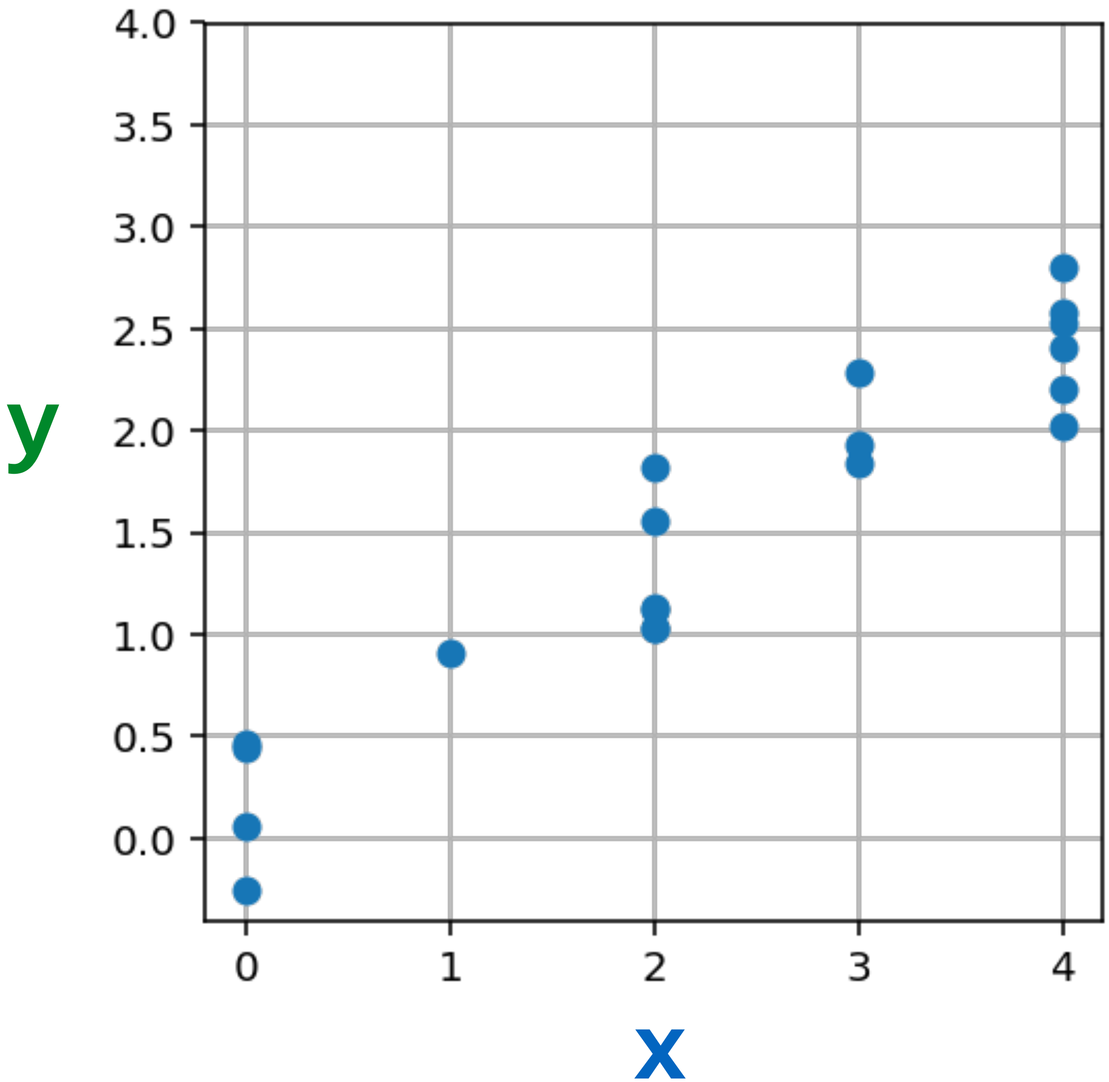
subject

stimulus

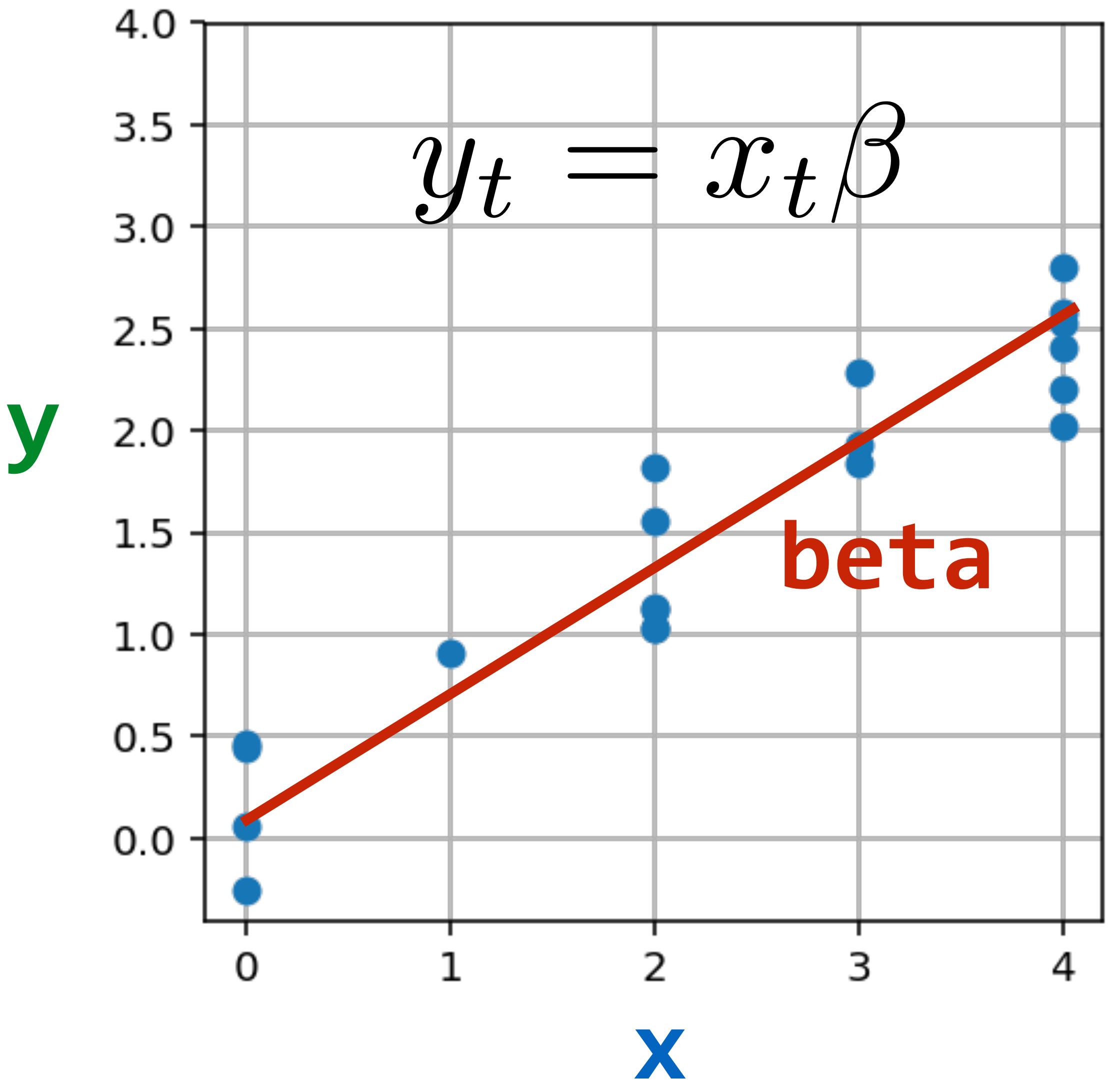




1D EXAMPLE



1D EXAMPLE



LEAST SQUARES

* we start solving this problem by specifying an **error function** (or **loss function**)

* the error function, $E(b)$, tells us how wrong the model is if we use the weight b

LEASTSQUARES

* the typical error function used for linear regression is the **squared error**

  * (leading the type of linear regression we’re talking about here to sometimes be called “linear least squares” or “ordinary least squares”)

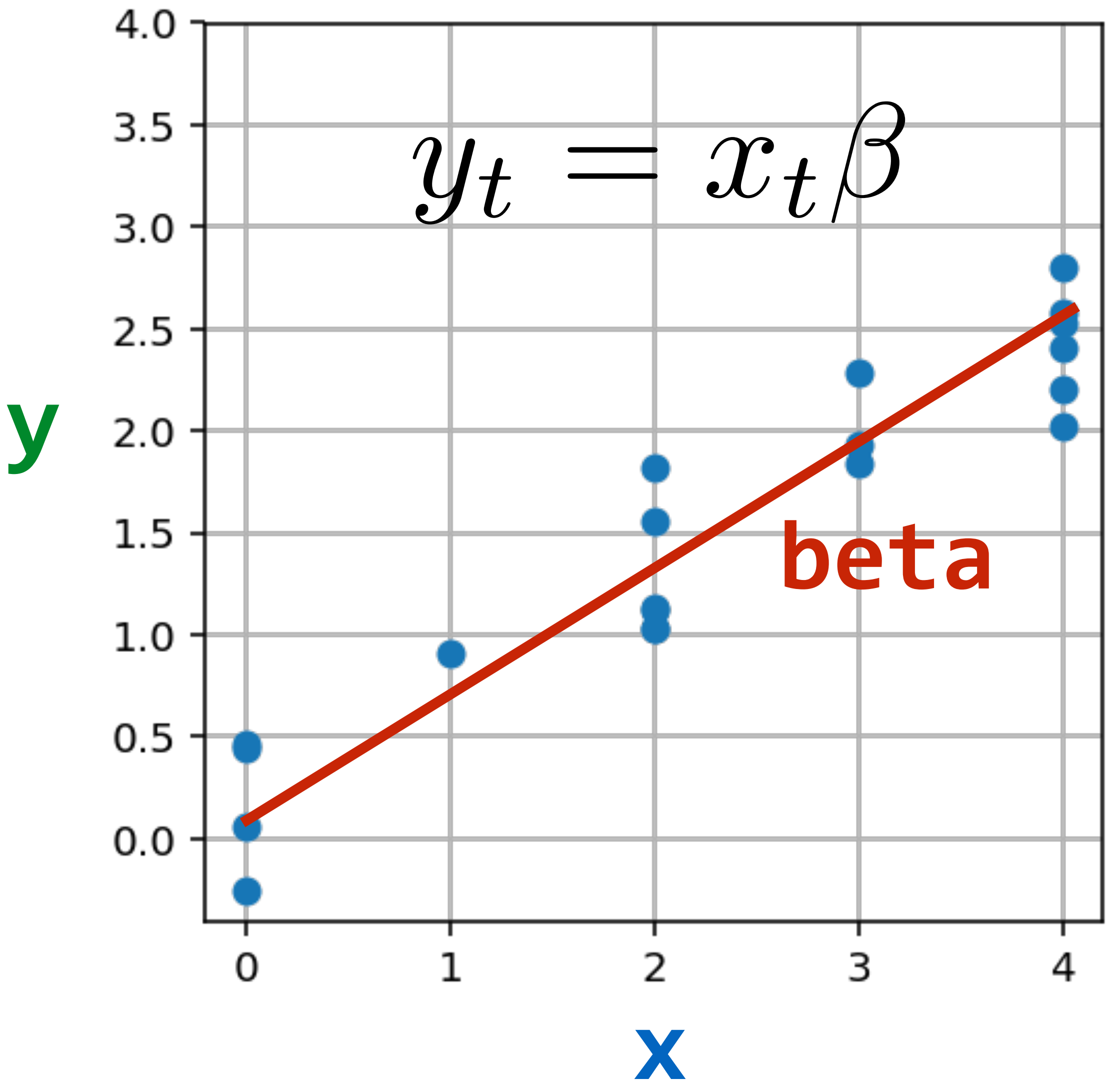
LEAST SQUARES

* it’s defined as the total squared difference between actual and predicted data

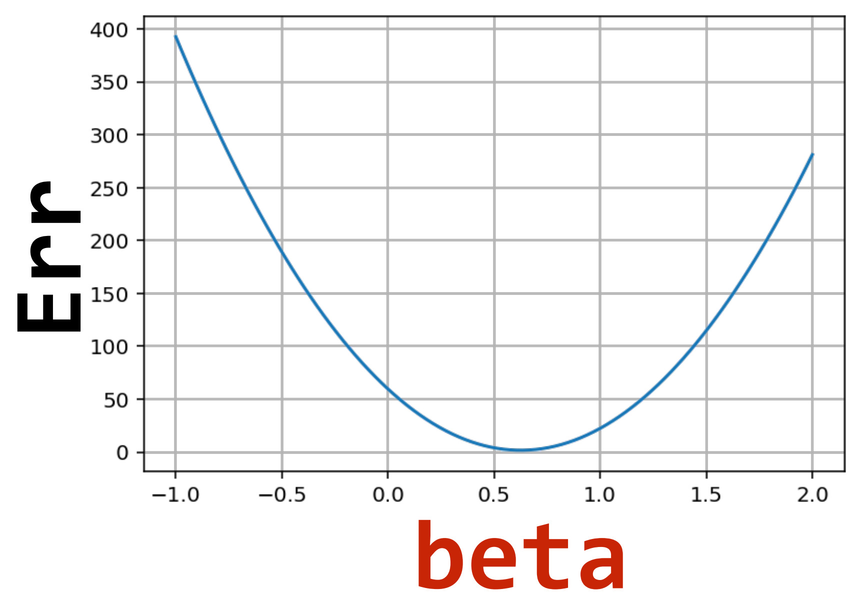
$$Err(\beta) = \sum_{t=1}^{T} (y_t - x_t \beta)^2$$

1D EXAMPLE

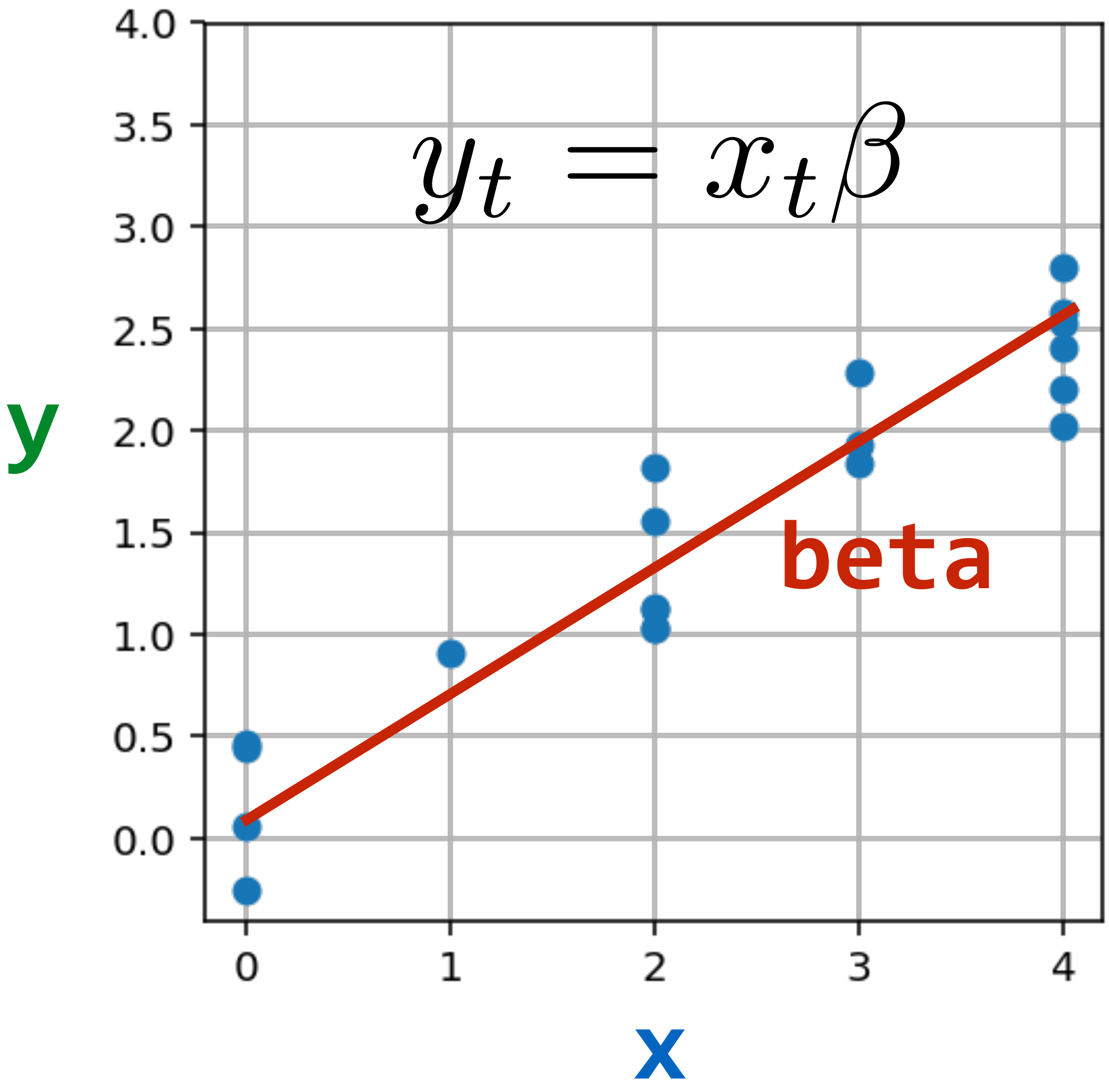
$$y_t = x_t \beta$$



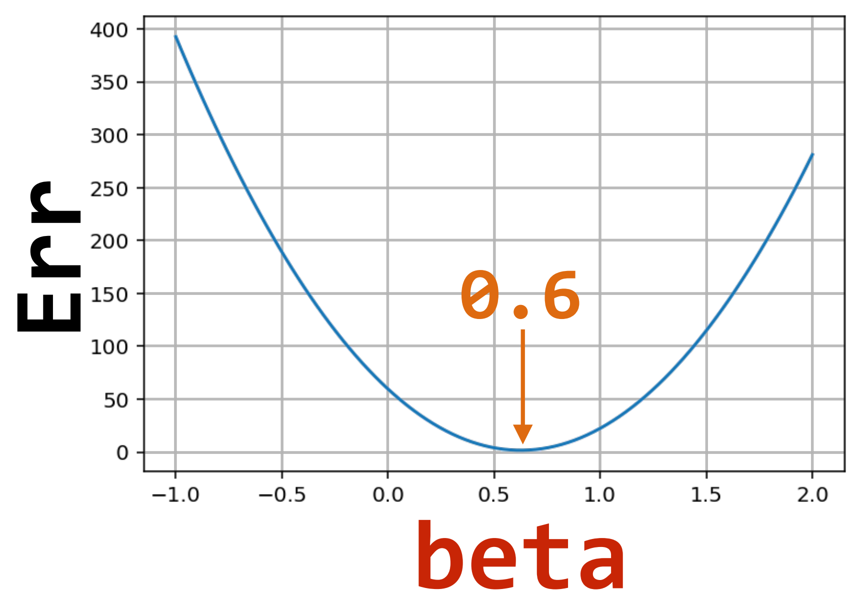
$$Err(\beta) = \sum_{t=1}^{T} (y_t - x_t \beta)^2$$



1D EXAMPLE



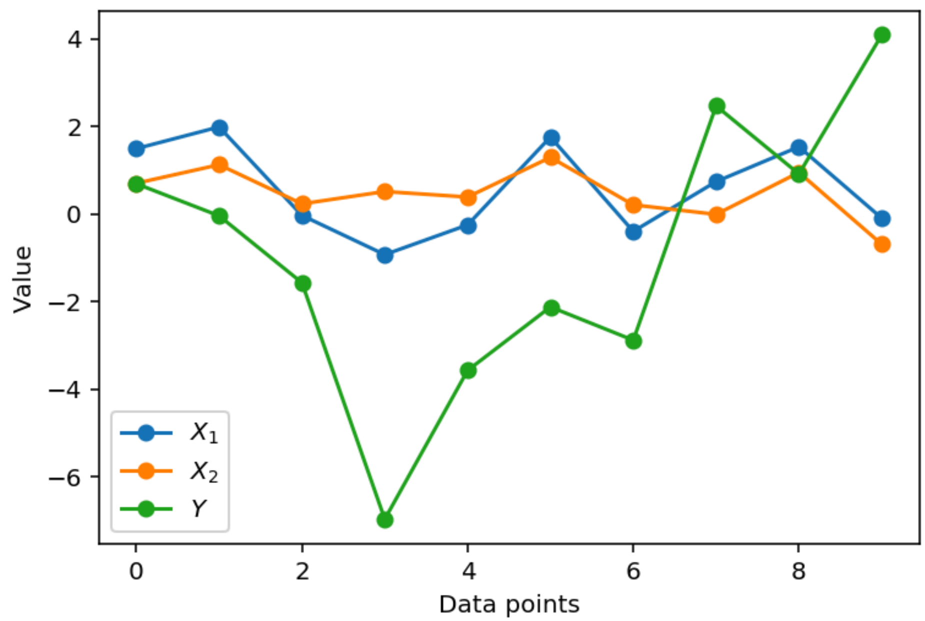
$$Err(\beta) = \sum_{t=1}^{T}(y_t - x_t\beta)^2$$



2D EXAMPLE

* now suppose we have two input variables, X1, and X2, and an output Y

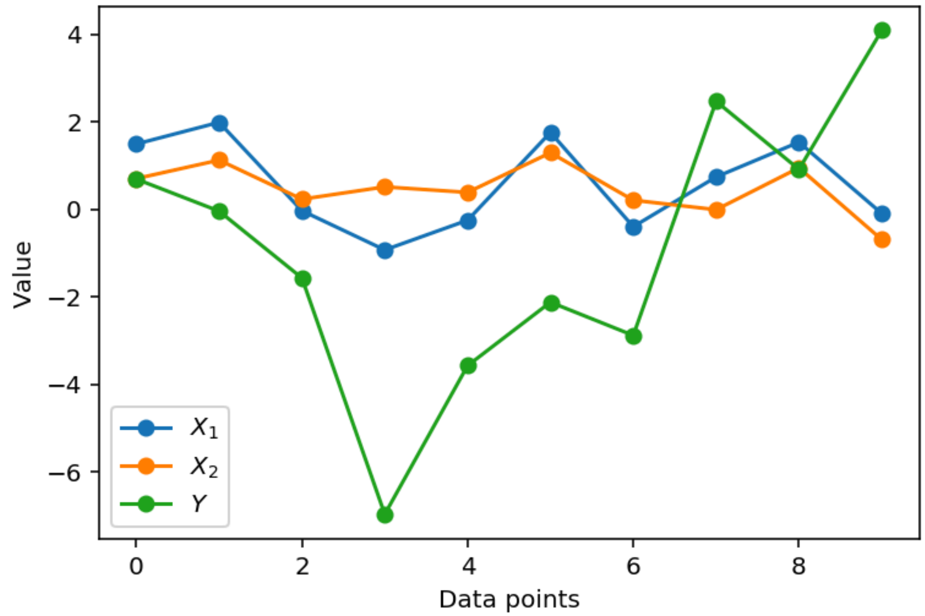
* we want to fit a model of the form:
  Y = X1*b1 + X2*b2



2D EXAMPLE

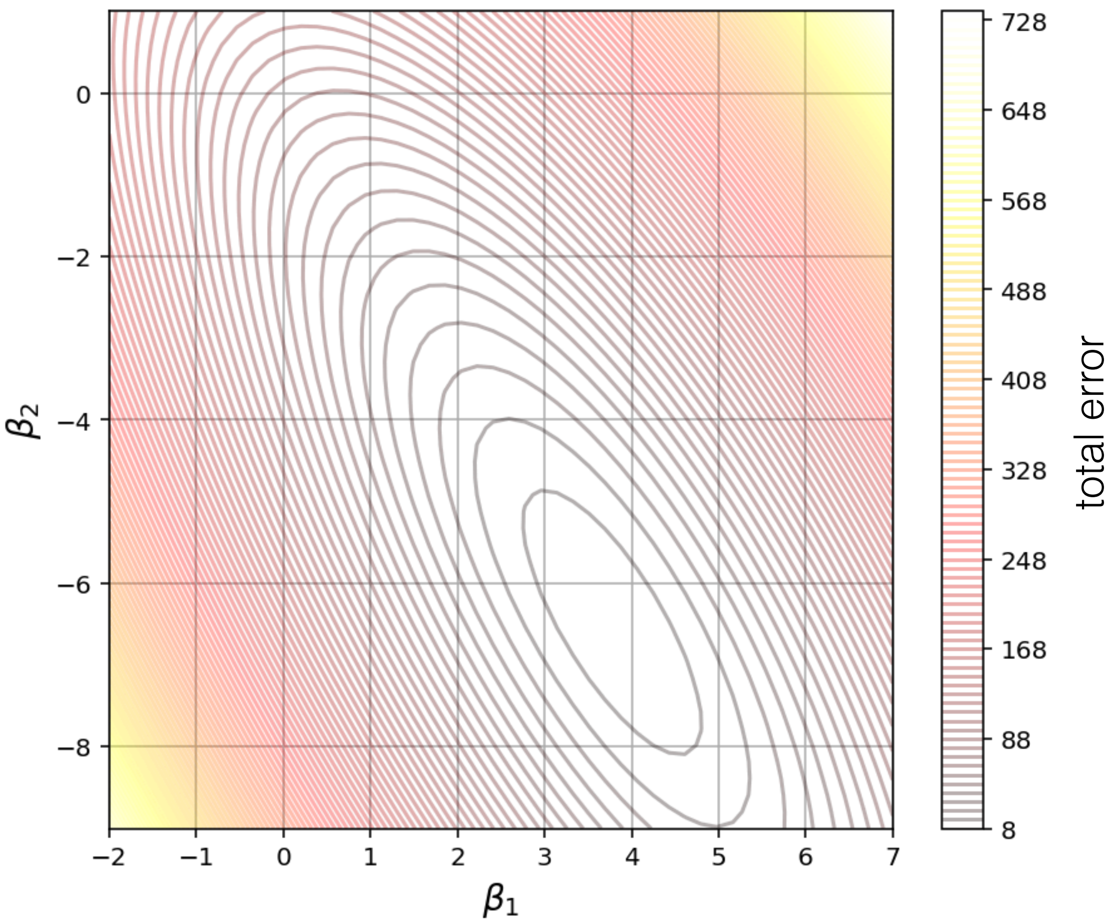
* Y = X1*b1 + X2*b2

* again b1 and b2 are called **weights** or **parameters**



2D EXAMPLE

* here the error function takes two variables, *E(b1, b2)*



LINEAR REGRESSION

* so how do we find the values of b1 and b2 that minimize the loss function?

LINEAR REGRESSION

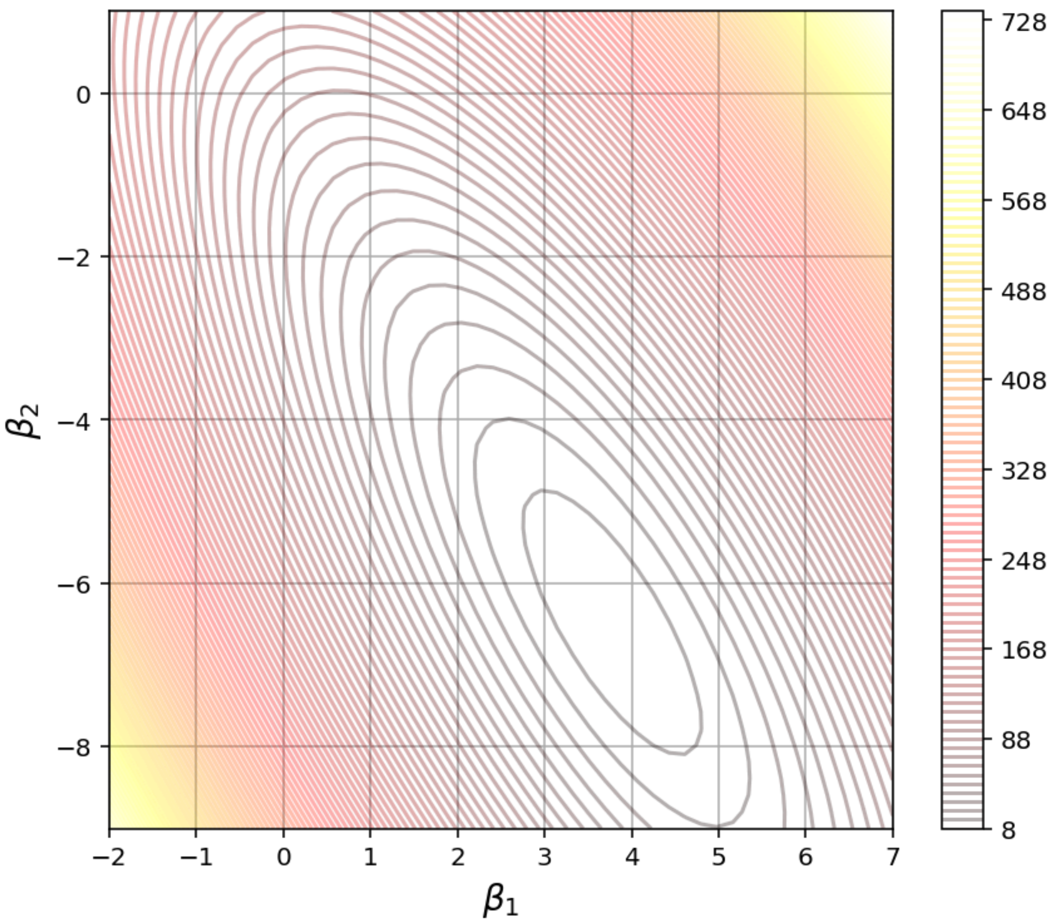
* we could do it “greedily”:

  * first find the b1 that minimizes error while keeping b2=0

  * then find the b2 that minimizes error while keeping b1 constant

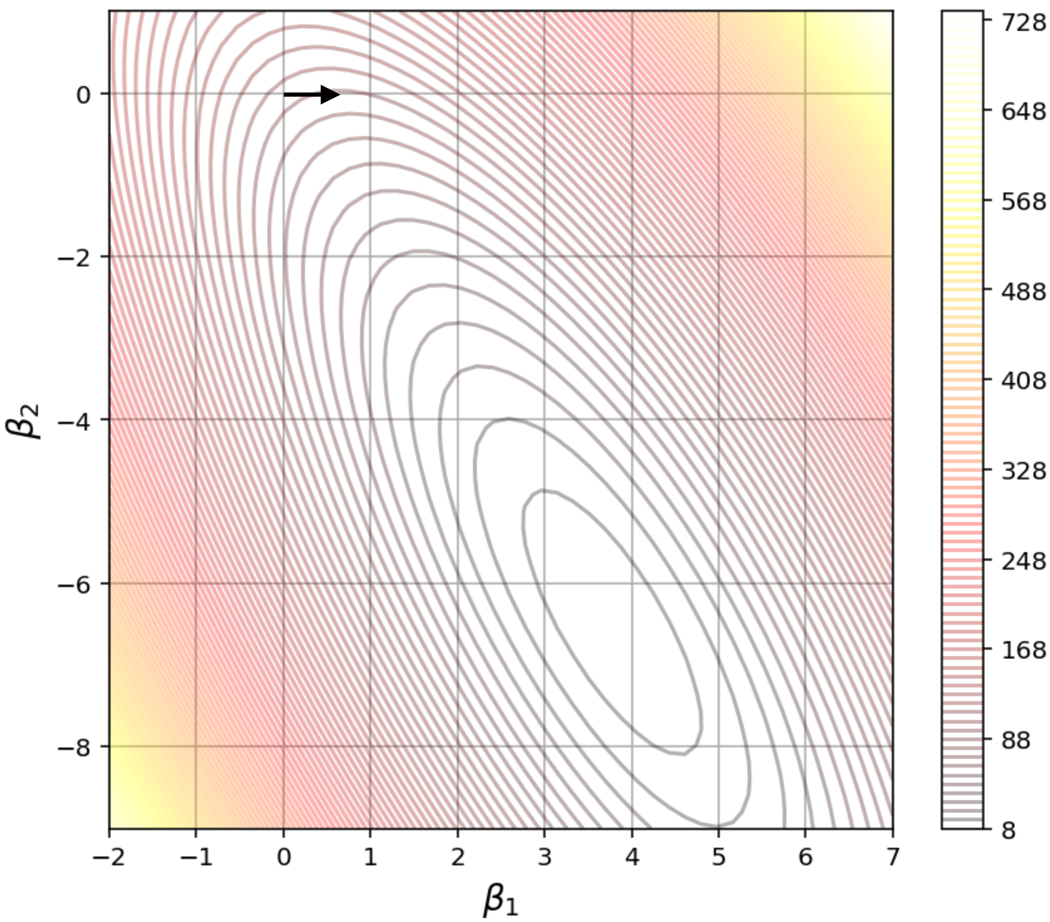
LINEAR REGRESSION

* this would not work well



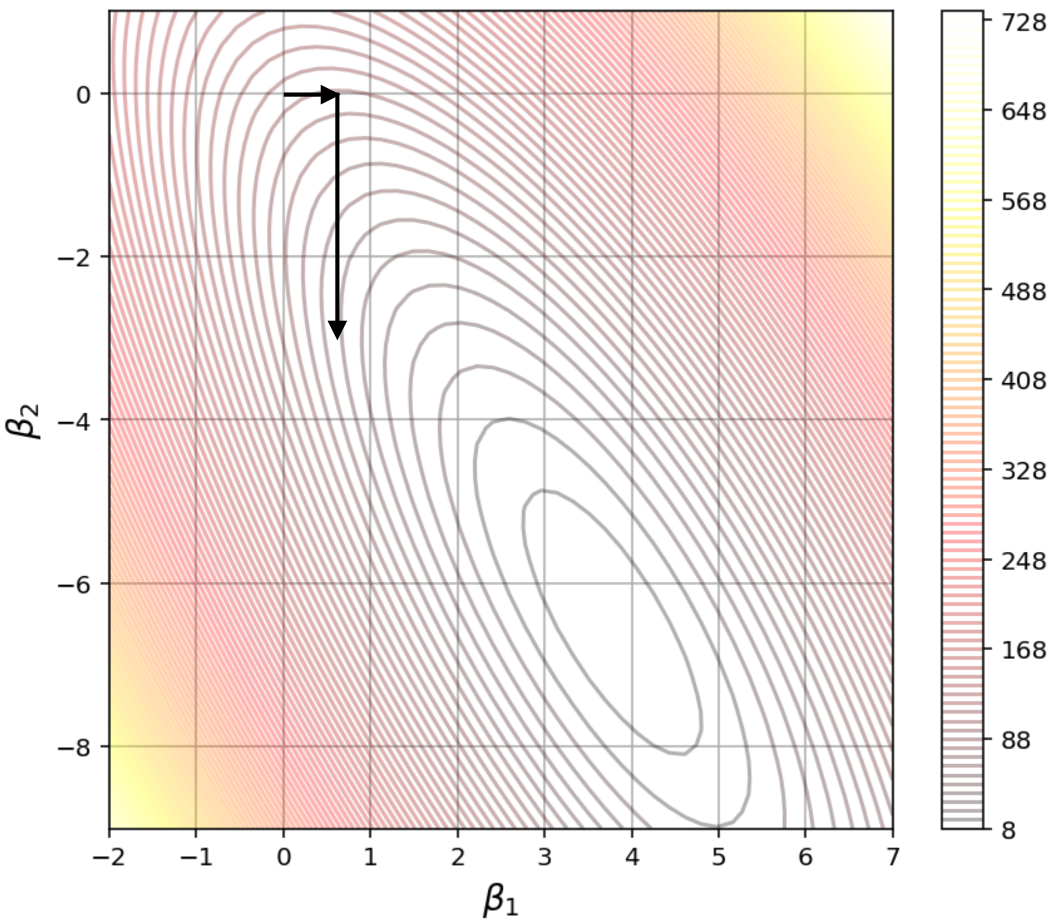
LINEAR REGRESSION

* this would not work well



LINEAR REGRESSION

* this would not work well



LINEAR REGRESSION

* the issue is that we need to account for how much X2 explains while fitting b1, while simultaneously accounting for how much X1 explains while fitting b2

LINEAR REGRESSION

* one way to do this is to mimic the "greedy" solution, but taking tiny steps

* which direction should each step point?

* we can find the "best" direction by computing the derivative (aka **gradient**) of the loss function with respect to b1 & b2

* this is called **gradient descent**

LINEAR REGRESSION

* (example)

LINEAR REGRESSION

* but there's another way to find the optimal b1 and b2

* what's the shape of the error function?

    *

LINEAR REGRESSION

* but there’s another way to find the optimal b1 and b2

* what’s the shape of the error function?

  * a parabola!

LINEAR REGRESSION

* there is an analytic solution to the minimum of a parabola!

$$\hat{\beta} = (X^\top X)^{-1} X^\top Y$$

END