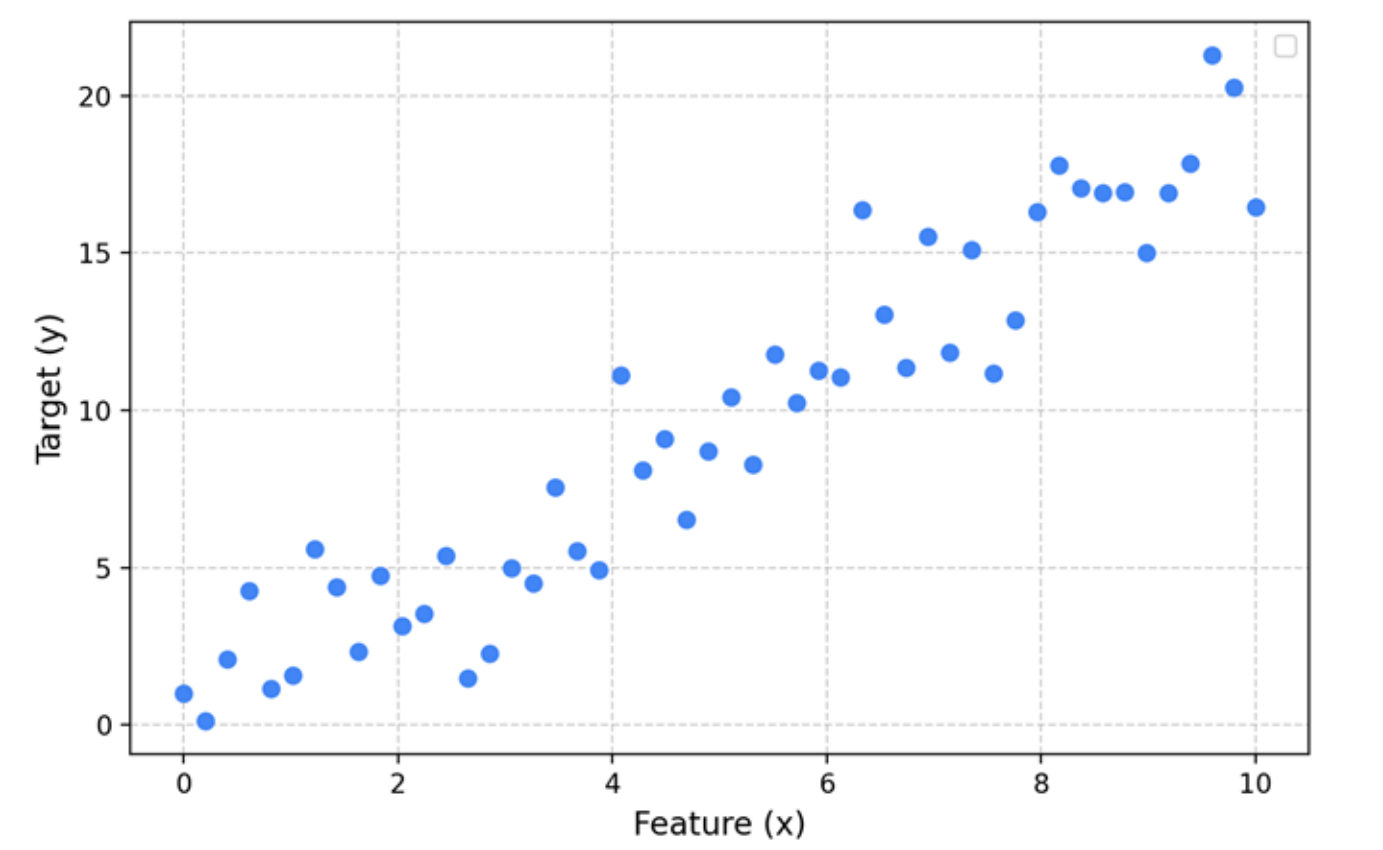
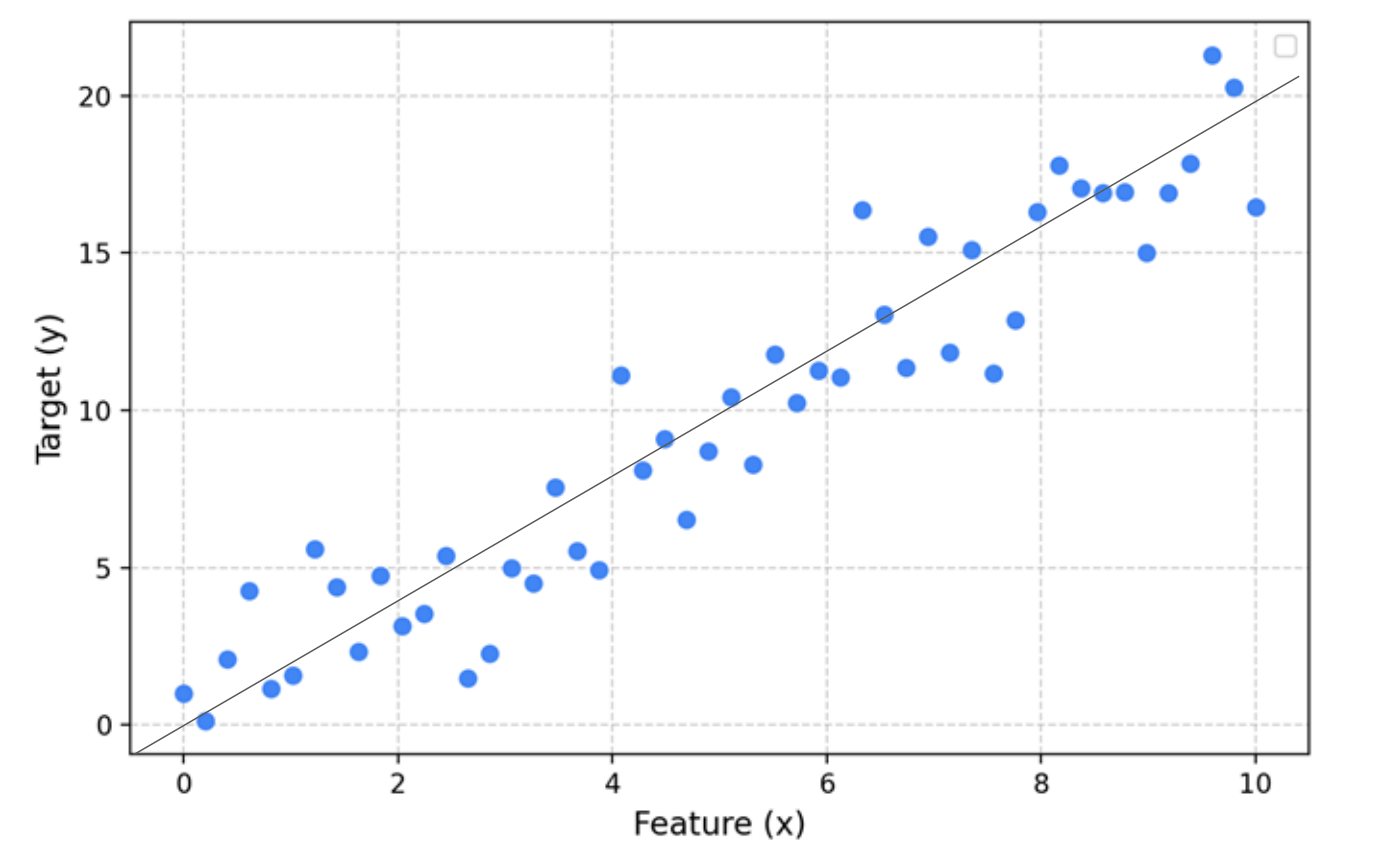


Intro to Machine Learning

Session 2 of 7

Recap of lecture 1:


Target (y)
Feature (x)
0
5
10
15
20
0
2
4
6
8
10


Target (y)
Feature (x)
0
5
10
15
20
0
2
4
6
8
10

Least Squares

$$L(w,b) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Residuals

The difference between the actual value and the prediction.

$e_i = y_i - \hat{y}_i$

Squared Error

Squaring stops negative/positive signs from cancelling out and amplifies large error.

${e_i}^2 = (y_i - \hat{y}_i)^2$

Solving It

Normal Equations

Exact closed-form mathematical solution in a single step.

$$w = (X^T X)^{-1} X^T y$$

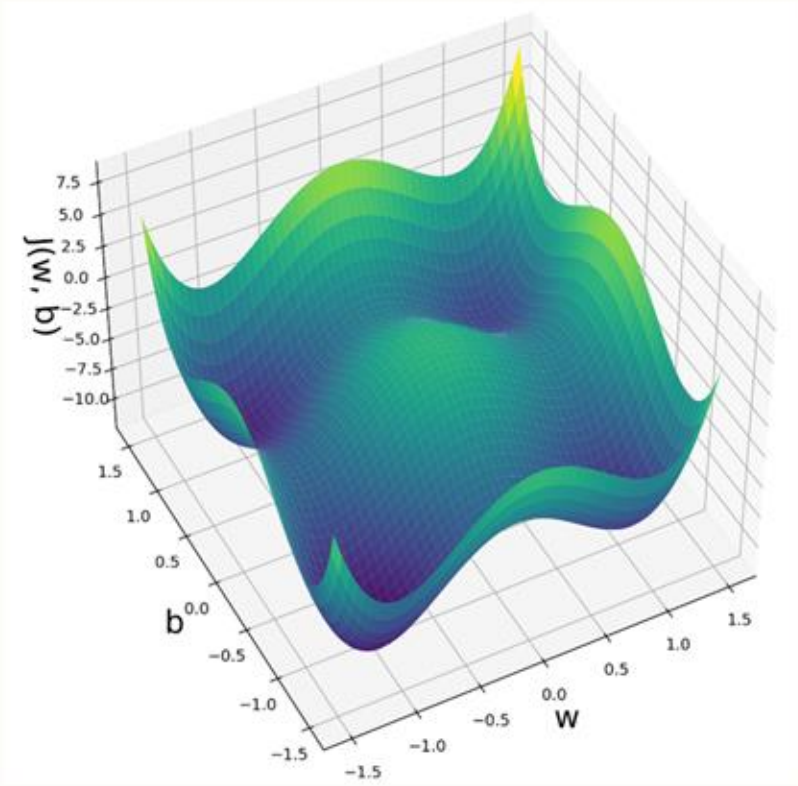
Gets very slow with millions of features.

Gradient Descent

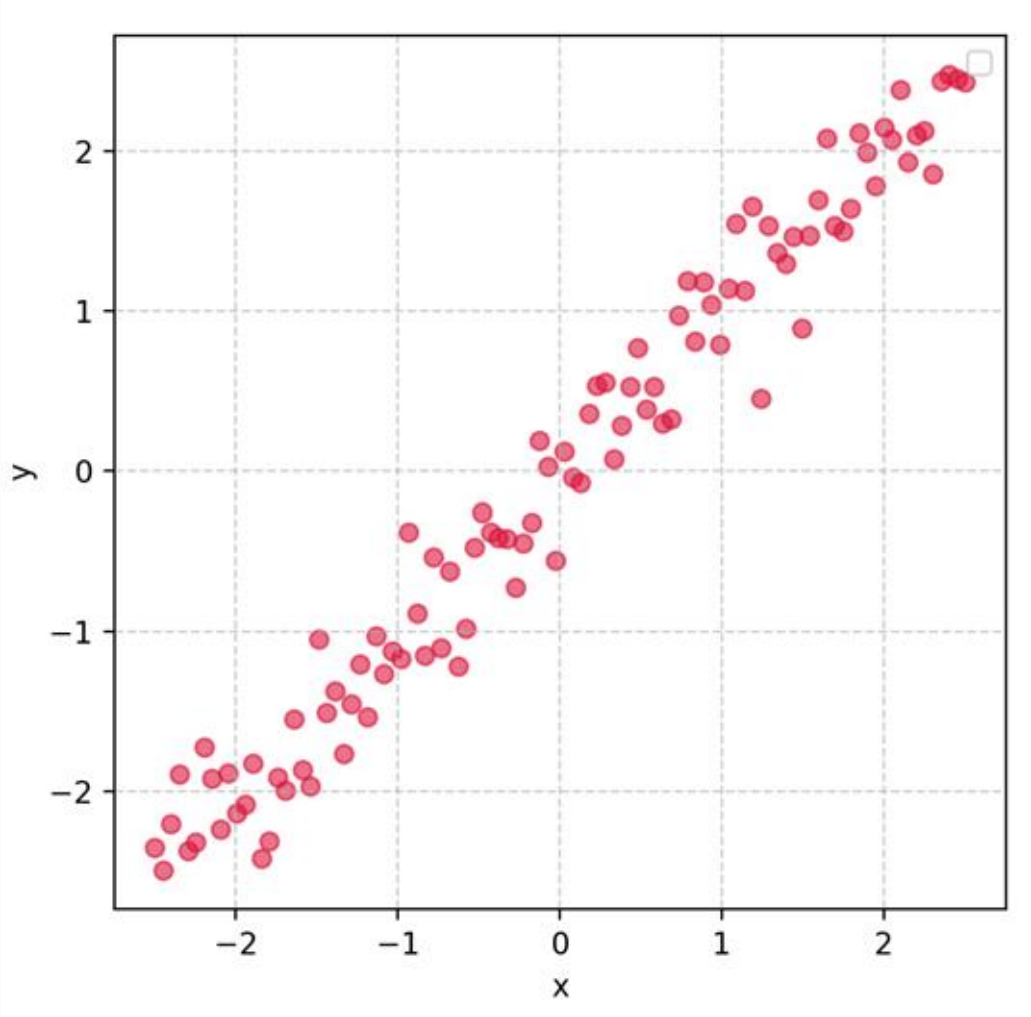
Repeatedly step downhill opposite to the gradient.

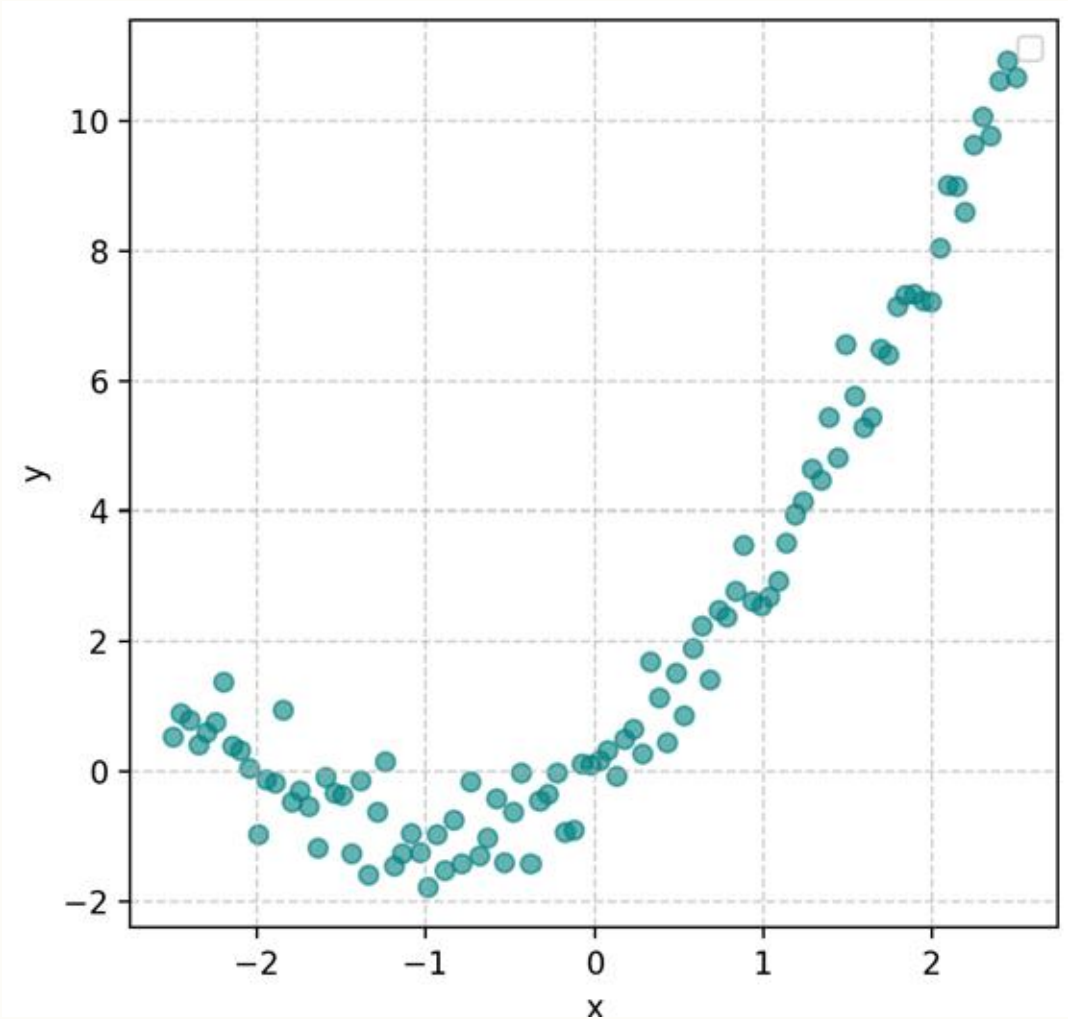
$$w = w - lr \frac{\partial}{\partial w} J(w,b) \qquad b = b - lr \frac{\partial}{\partial b} J(w,b)$$

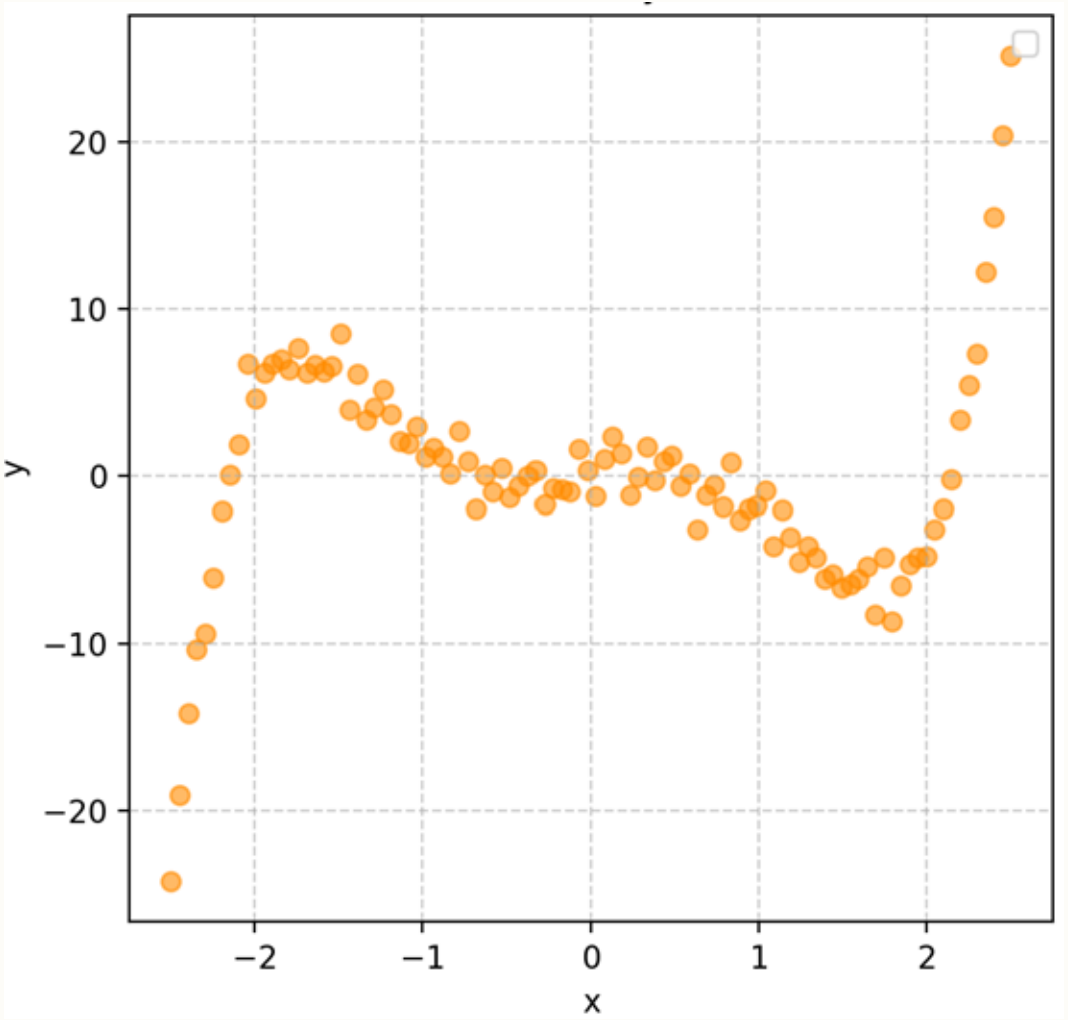
Scales efficiently to massive datasets.



Where does this fall short?


2
1
0
−1
−2
y
−2
−1
0
1
2
x


10
8
6
4
2
0
−2
y
−2
−1
0
1
2
x


20
10
0
−10
−20
y
−2
−1
0
1
2
x



Multivariate Regression

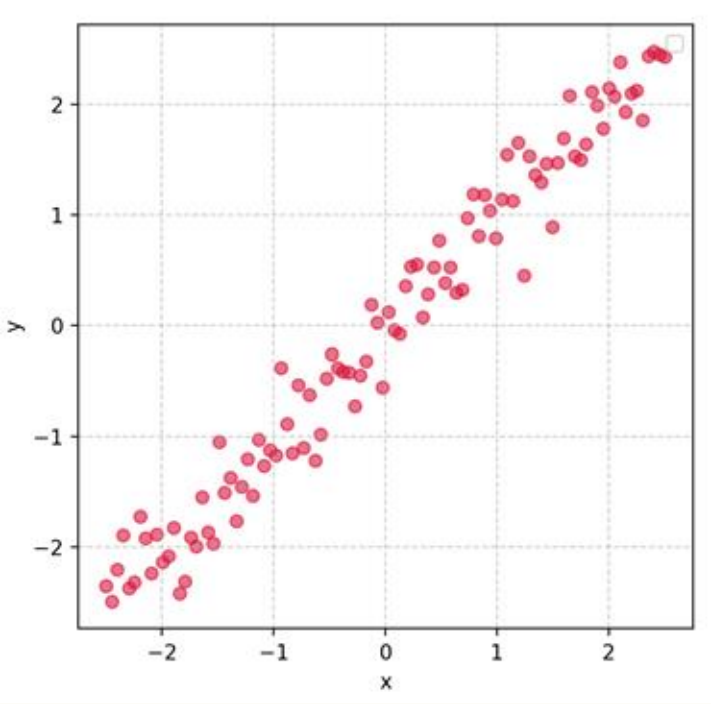
Beyond One Dimension

ONE FEATURE

$$\hat{y} = wx + b$$

MANY FEATURES

$$\hat{y} = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + b = w^T x + b$$

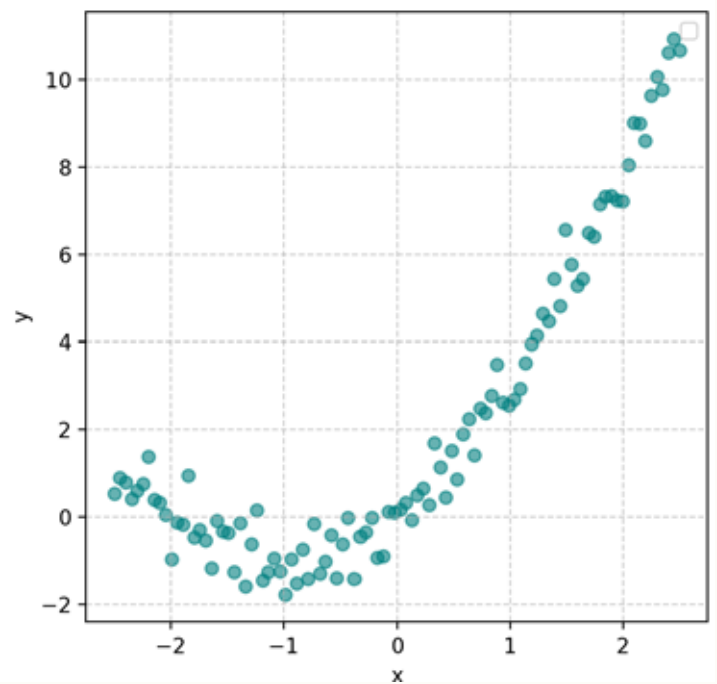


Absorbing the Bias

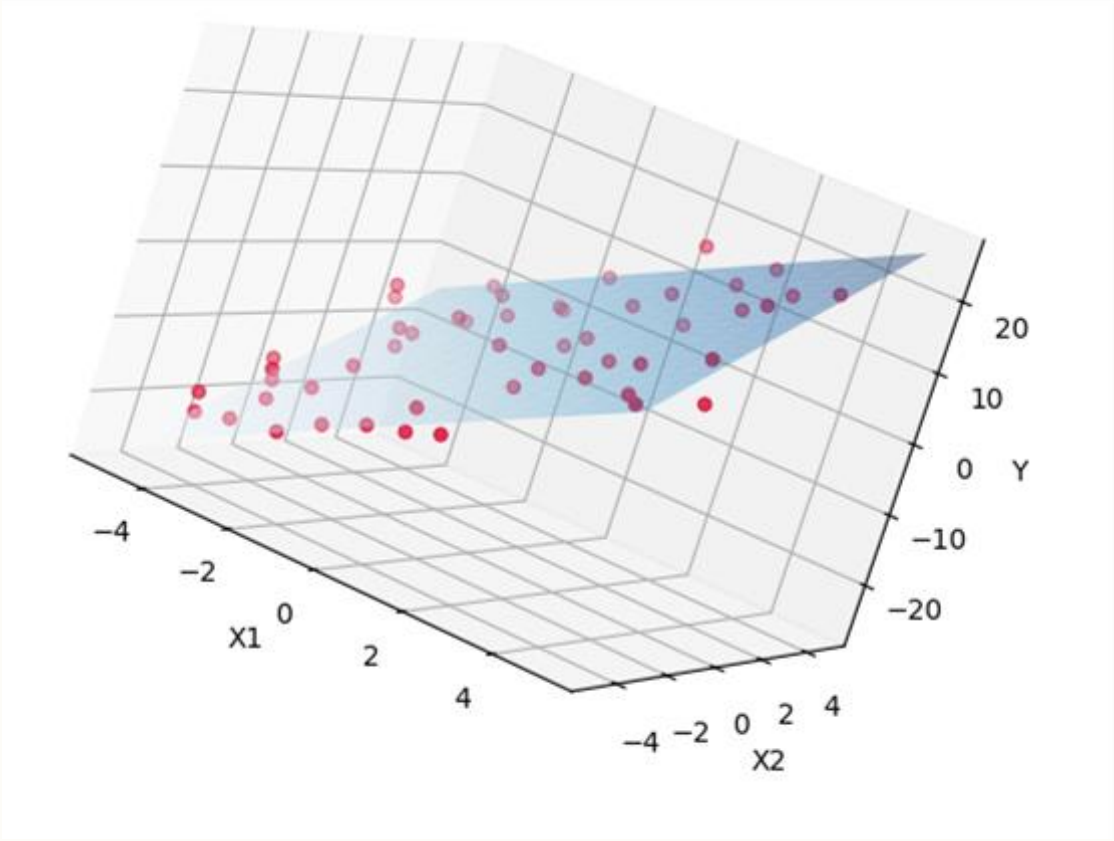
By setting a constant dummy feature $x(0) = 1$, the intercept b is absorbed into the weight vector as $w(0)$. This simplifies the prediction equation.

$$\hat{y} = w^T x$$

Beyond One Dimension



Beyond One Dimension



Can a the model fit a curve?

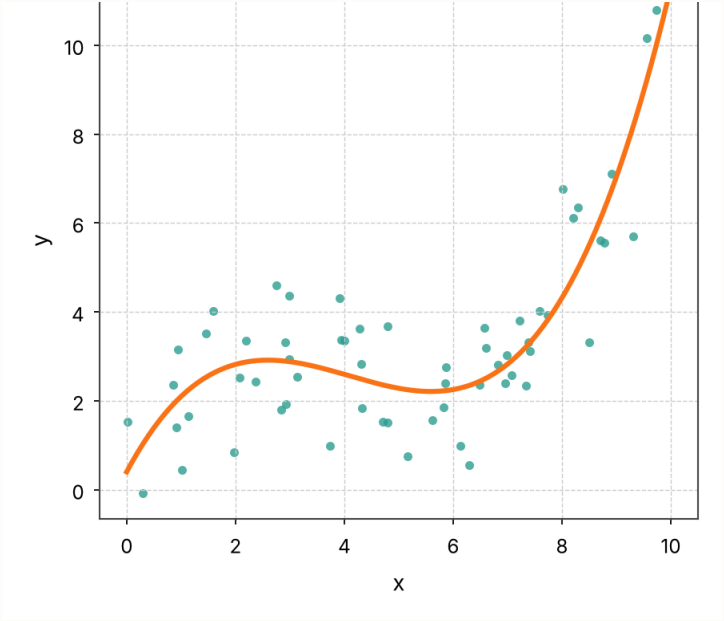
Polynomial Features

ONE INPUT, MANY FEATURES

$$\hat{y} = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

AS A WEIGHT VECTOR

$$\hat{y} = \mathbf{w}^T \mathbf{x}$$



The Design Matrix

$$\underbrace{\begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & x_n^3 \end{bmatrix}}_{X \in \mathbb{R}^{n \times 4}} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} \approx \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Squared Error

Same loss as before, now over four weights.

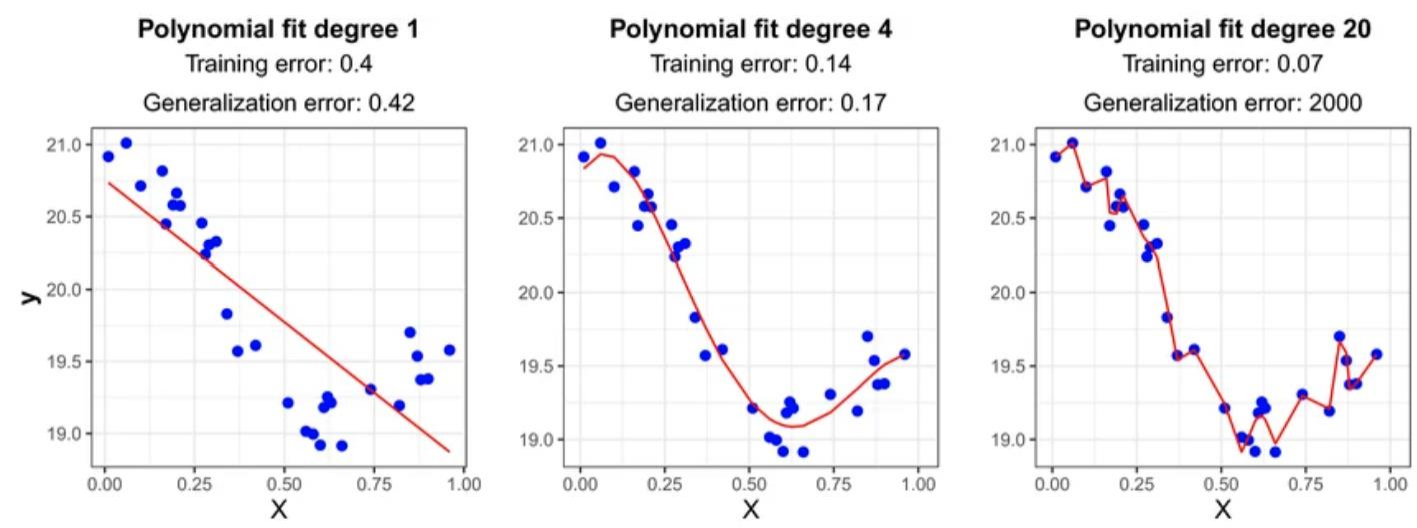
$$L(\mathbf{w}) = \frac{1}{2n} \|X\mathbf{w} - \mathbf{y}\|^2$$

Normal Equations

Each power of x is just another column of X.

$$\mathbf{w} = (X^T X)^{-1} X^T \mathbf{y}$$

Choosing the Degree



Degree 1

Underfits: a straight line cannot bend.

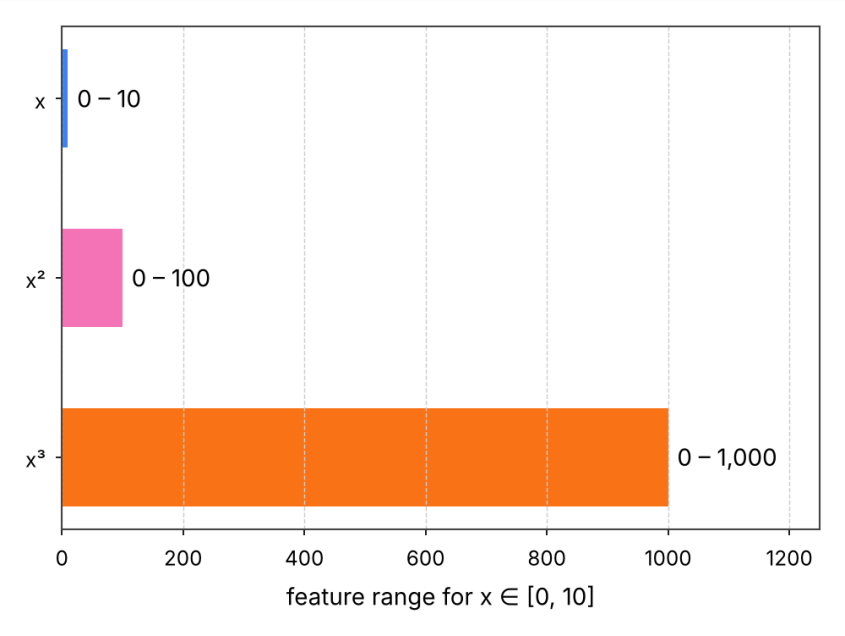
Degree 4

Tracks the curve well, and is the most accurate model

Degree 3

Overfits: Tracks every point perfectly, with the lowest error, but generalises terribly.

Why Scale Features?



The Problem

For x between 0 and 10, x^3 is 100 times larger than x. The error calculation is almost 100% dominated by x^3. The algorithm completely ignores x and x^2, effectively throwing away those features.

Feature Scaling

Standardization

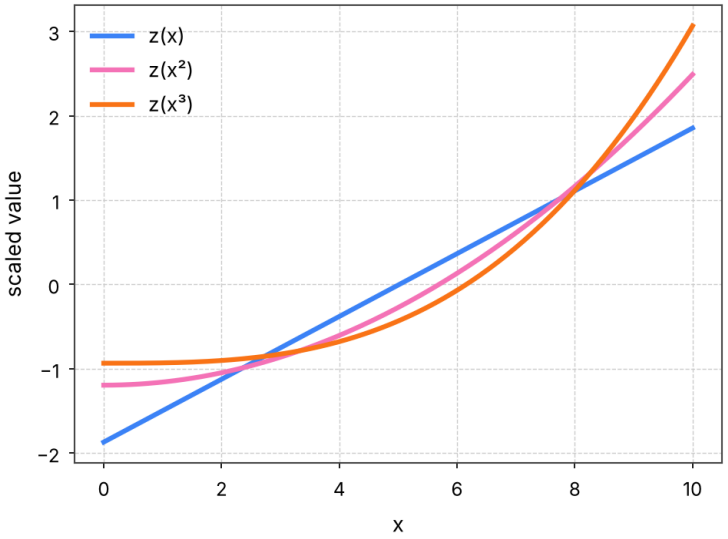
Mean 0, standard deviation 1.
The usual default.

$$z = \frac{x - \mu}{\sigma}$$

Min–Max

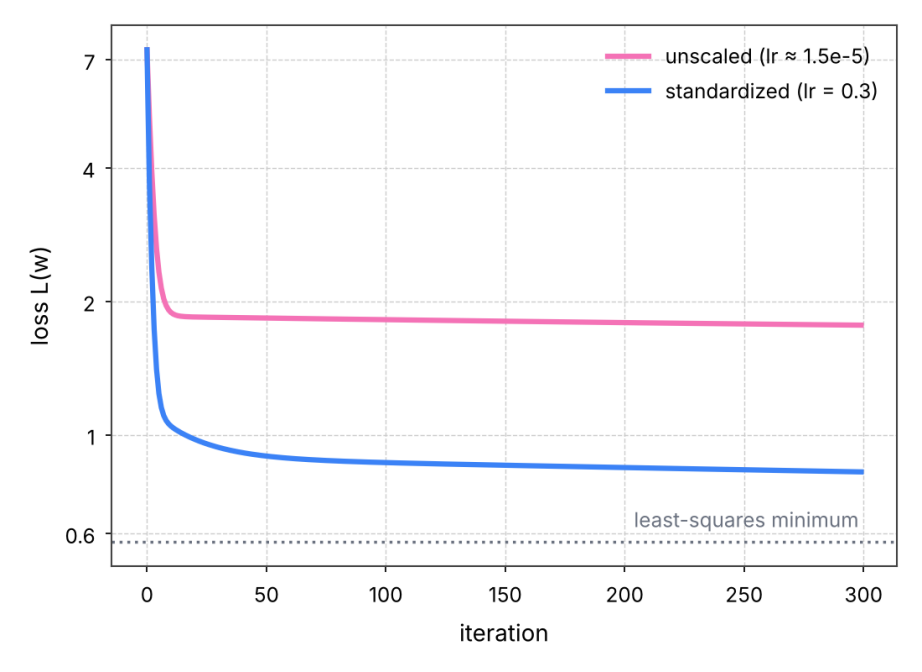
Squeezes each column into the range 0 to 1.

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$



Rules of thumb: scale x, x^2 and x^3 separately, leave the column of 1s alone, and compute μ and σ on training data only, then reuse them at prediction time.

Scaled Gradient Descent



Unscaled

Learning rate must stay below 1.7×10^{-5}. Loss is still 1.77 after 300 steps.

Standardized

A learning rate of 0.3 is stable. Loss is 0.83 after 300 steps.

Key Takeaways

1

Add columns

Treat x, x^2 and x^3 as separate features.

2

Solve as before

Use the normal equations or gradient descent on the design matrix X.

3

Scale first

Standardize each power of x, but not the bias column, so training is fast and stable.