CS60050
Machine Learning

Linear Regression Part 1

Sudeshna Sarkar Somak Aditya
Department of CSE, IIT Kharagpur
August 9, 2023

Dataset of living area and price of houses in a city

Living area ($feet^2$)	Price (1000\$s)
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮

← Training Set

Predict → $\boxed{5000}$ → $\boxed{?}$

Regression

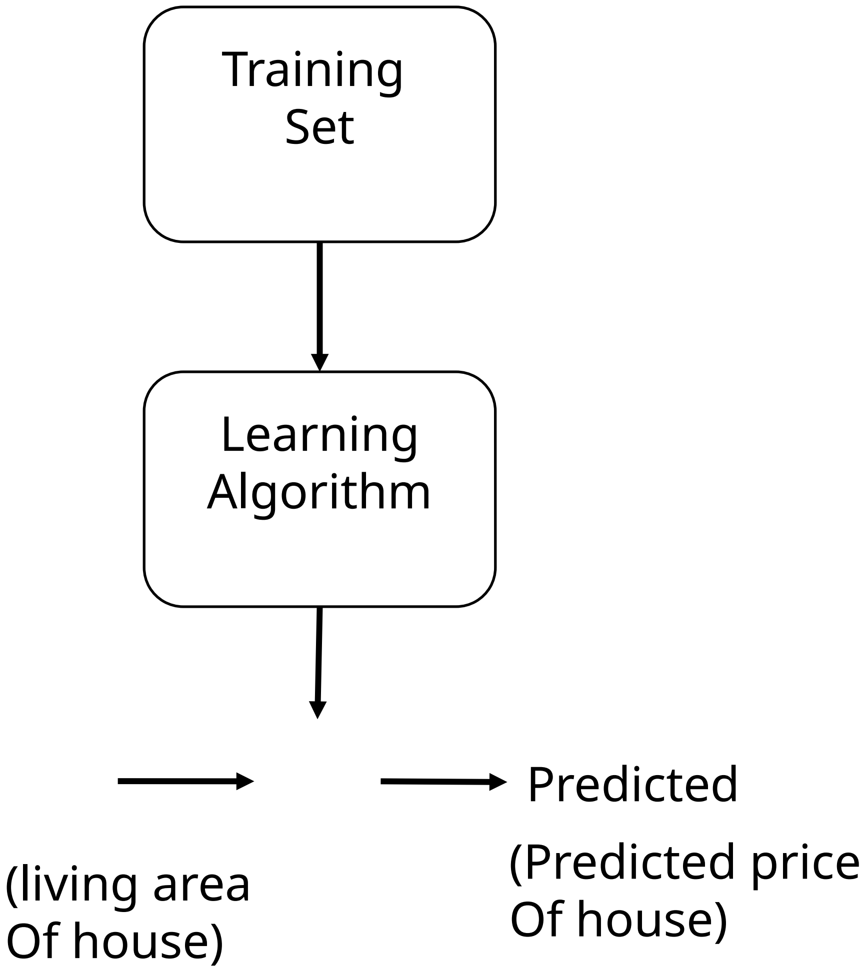
= number of **training examples**

= input variables / features

= output variables / "target" variables

- i^{th} training example of the training set

How to use the training set?



- Learn a function , so that is a good predictor for the corresponding value of y
- hypothesis function

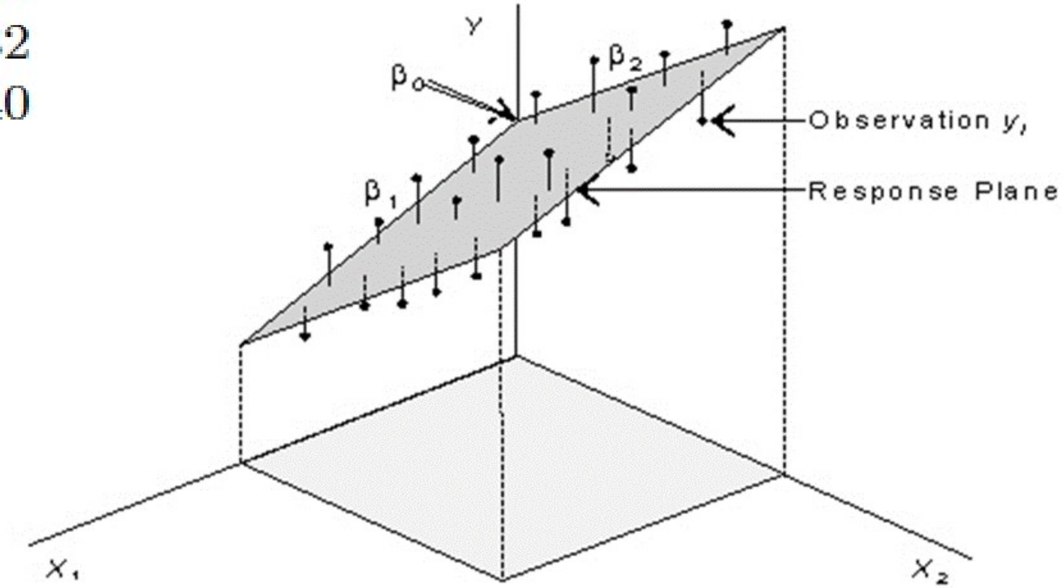
How to represent hypothesis? (linear?)

- are **parameters**
- : vector of all the parameters

- We assume
  - is a linear function of
- How to learn the values of the parameters ?

Multivariate Regression

Living area (feet^2)	#bedrooms	Price (1000$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮

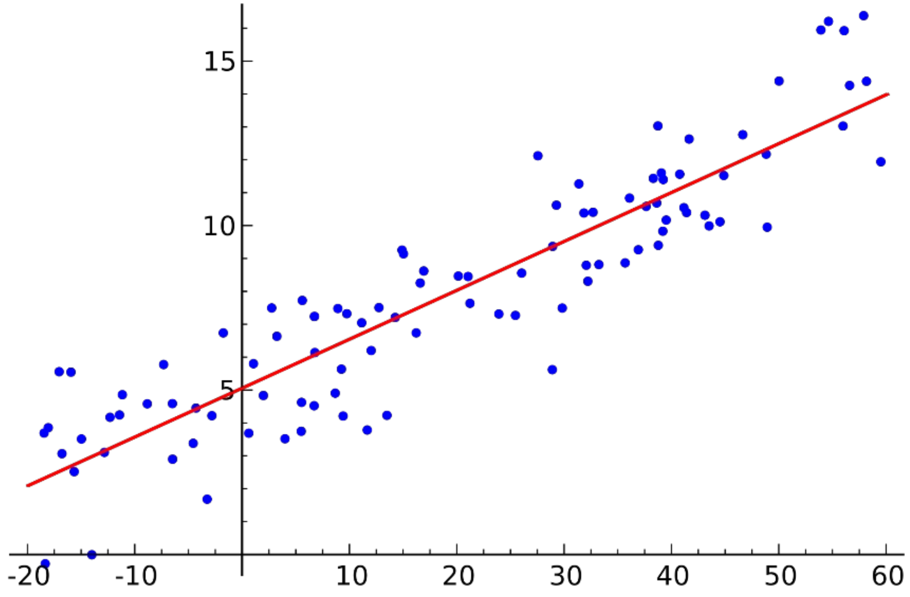


- features
- training examples
- : th training example

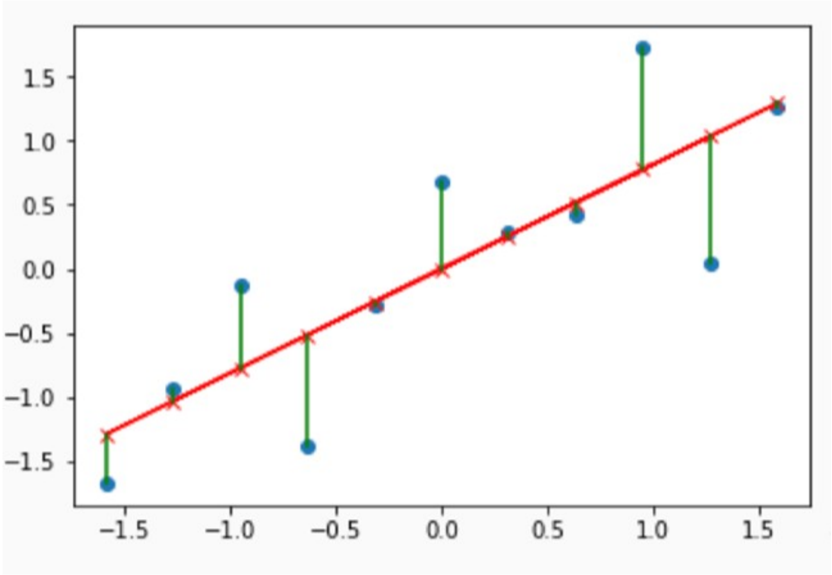
Intuition of hypothesis function

Two equivalent questions:

1. Which is the best straight line to fit the data?
2. How to learn the values of the parameters ?



Cost function



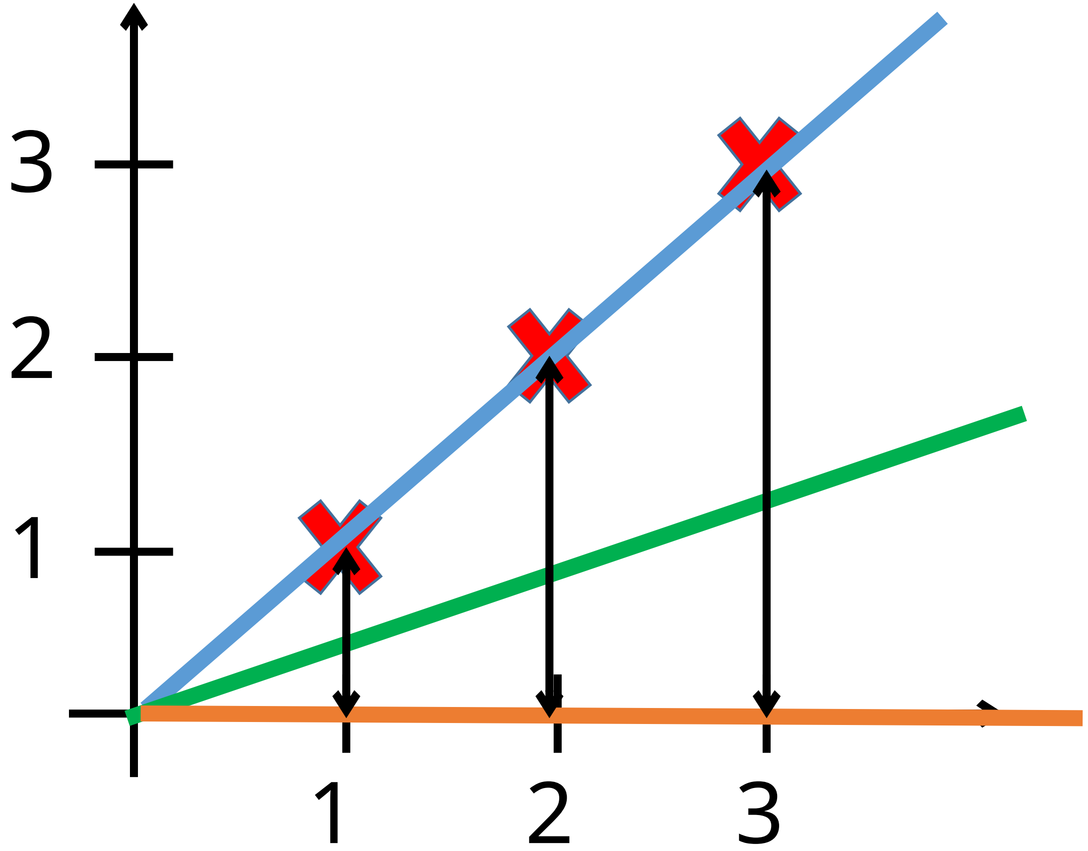
prediction error for th training example

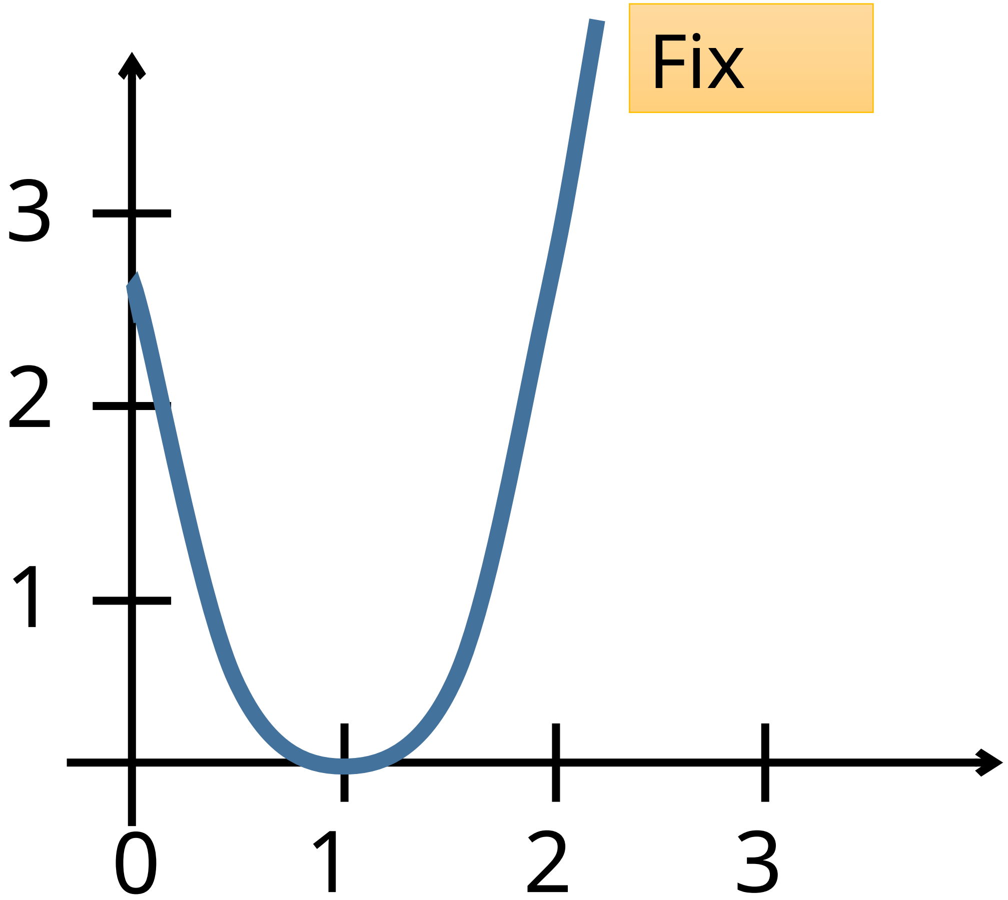
Choose parameters so that

is minimized

: function of for fixed

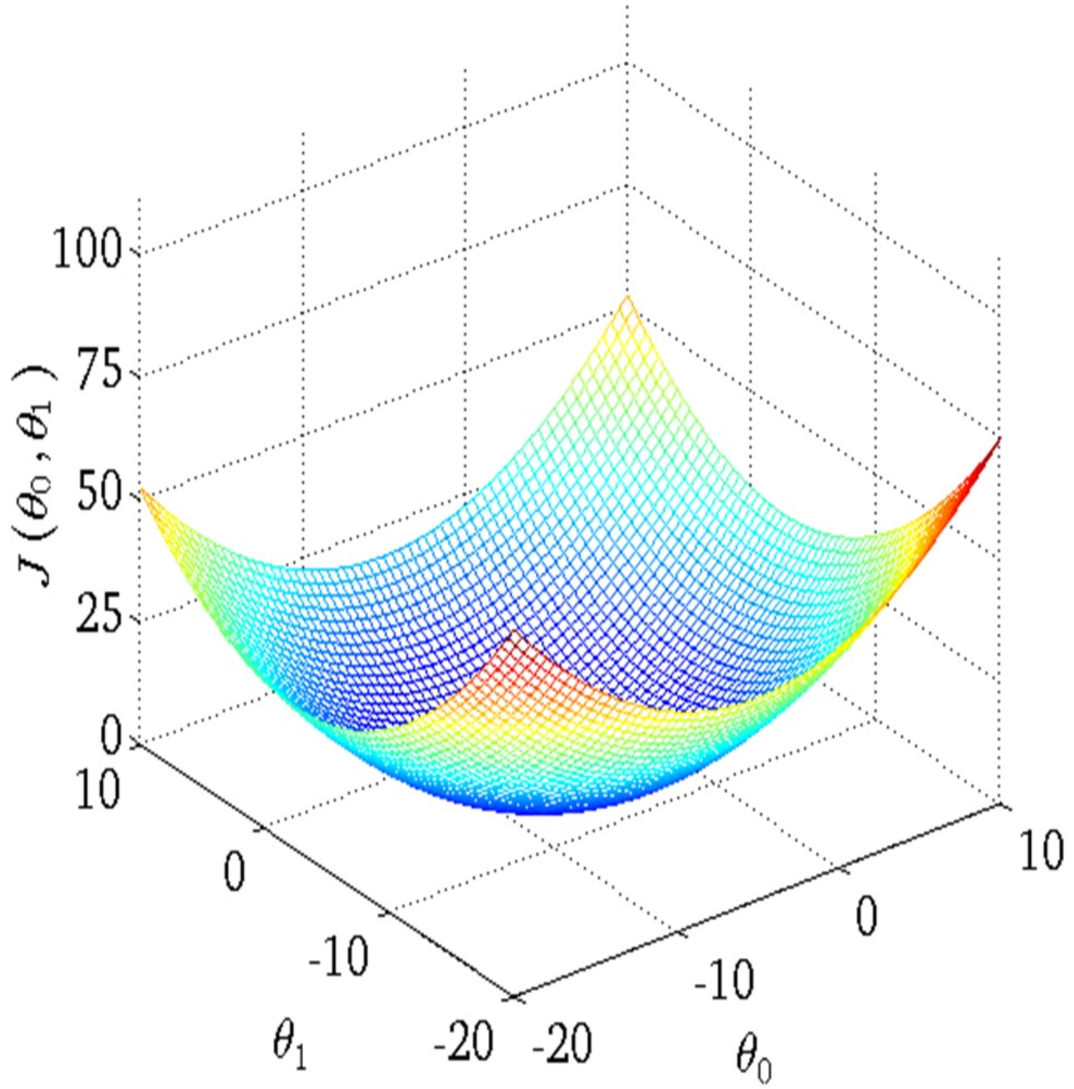
, function of





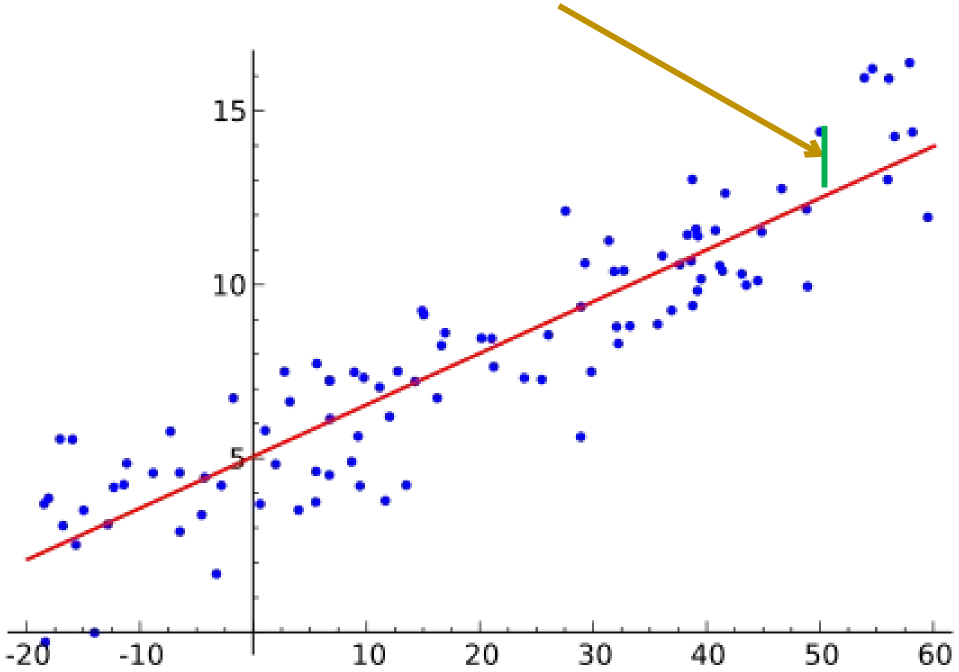
Cost Function

When loss is a function of
both θ_0 and θ_1



Linear Regression

The loss is the squared loss



Data pairs are the blue points.
The model is the red line.

Optimization objective: Find model parameters that will minimize the loss.

Linear Regression

The total loss across all points is

We want the optimum values of that will minimize the sum of squared errors. Two approaches:

1. Analytical solution via mean squared error
2. Iterative solution via MLE and gradient ascent

Learning as Optimization Problem

Hypothesis:

Parameters:

Cost

Function:

Goal:

Linear Regression: Analytic Solution

Since the loss is differentiable, we set

and

We want the loss-minimizing values of , so we set

These being linear equations of θ, have a unique closed form solution

Multivariate Linear Regression

Define

Cost Function:

Multivariate Linear Regression

Multivariate Linear Regression

- Equating the gradient of the cost function to 0,

This gives a closed form solution, but another option is to use iterative solution

Partial derivatives

- Let be a multivariate function with *n* variables
  - The mapping is
- The ***partial derivative*** of *y* with respect to its i^{th} parameter is
- To calculate , we can treat   as constants and calculate the derivative of *y* only with respect to
- For notation of partial derivatives, the following are equivalent:

Multidimensional derivative: Gradient

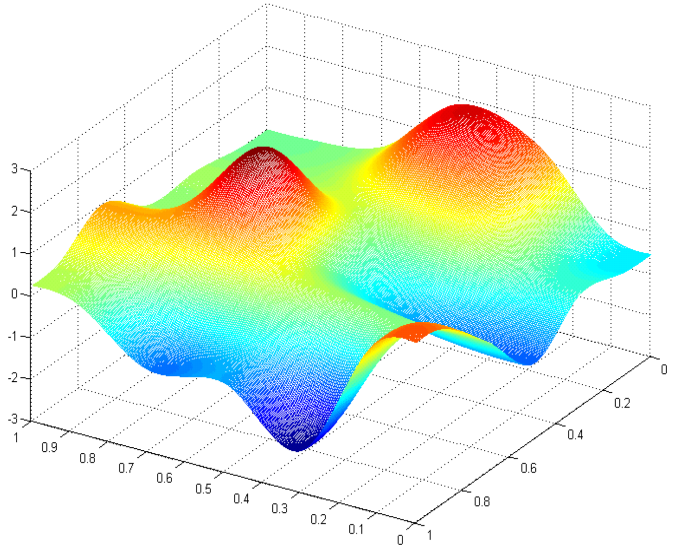
Gradient vector: The gradient of the multivariate function with respect to the *n*-dimensional input vector , is a vector of *n* partial derivatives

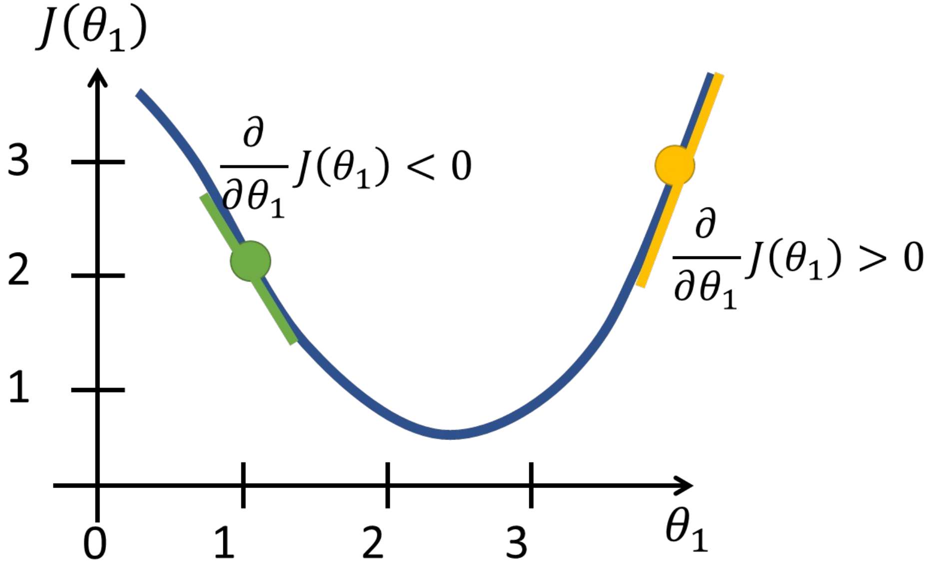
- In ML, the gradient descent algorithm relies on the opposite direction of the gradient of the loss function with respect to the model parameters for minimizing the loss function

Minimizing cost function & Gradient Descent

Minimizing function

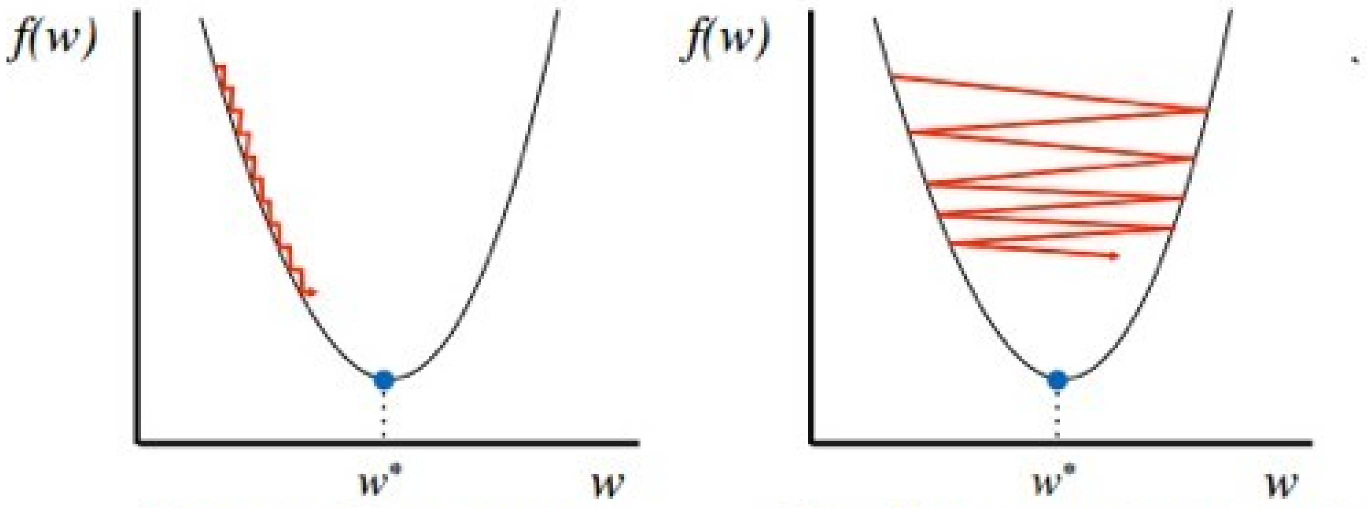
- Start with some
- Keep changing to reduce
- until we end up at a minimum





Step Size

- Determines how quickly training loss goes down; hence "learning rate"



Too small: converge very slowly

Too big: overshoot and even diverge

Computing partial derivatives

Repeat until convergence{

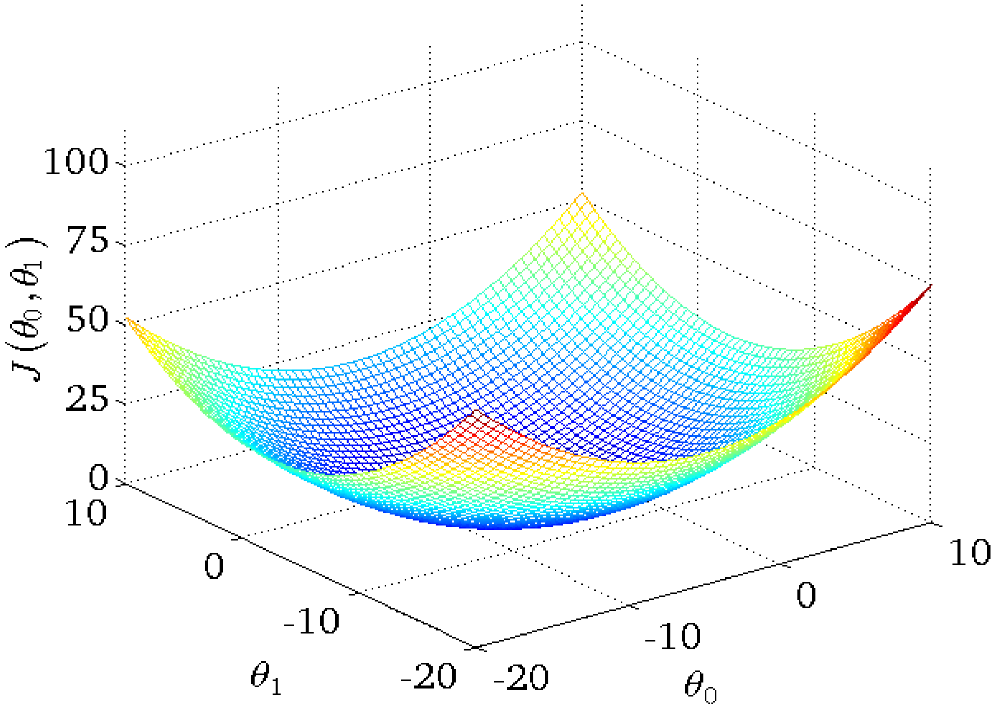
Equivalently

}

Convergence

- The cost function in linear regression is always a convex function – always has a single global minimum

- So, gradient descent will always converge



Batch gradient descent

"Batch": Each step of gradient descent uses all the training examples

: Number of training examples

Repeat until convergence{

}