



CS60050: Machine Learning

Autumn 2024

Sudeshna Sarkar

Linear Regression Part 1

26-27 July 2024



Dataset of living area and price of houses in a city

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

← Training Set

Predict → 5000 → ?

Regression

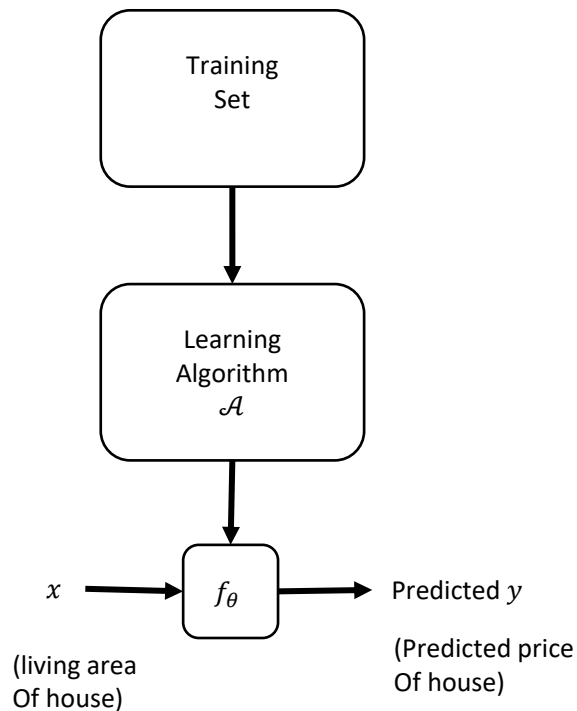
m = number of **training examples**

x_i = input variables / features

y_i = output variables /
"target" variables

(x_i, y_i) - i^{th} training example of the
training set

How to use the training set?



- Learn a function $f(x)$, so that $f(x)$ is a good predictor for the corresponding value of y
- f : hypothesis function

How to represent hypothesis? (linear?)

$$\hat{y} = f_{\theta}(x) = \theta_0 + \theta_1 x$$

- θ_i are **parameters**
- θ : vector of all the parameters
- We assume
 - y is a linear function of x
- How to learn the values of the parameters θ_i ?

Multivariate Regression

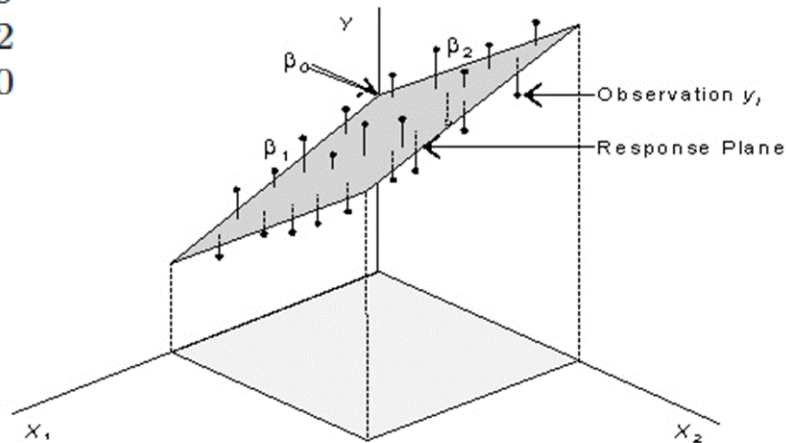
Living area (feet ²)	#bedrooms	Price (1000\$)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
\vdots	\vdots	\vdots

n features

m training examples

$(x^{(i)}, y^{(i)})$: i th training example

$$y = f_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

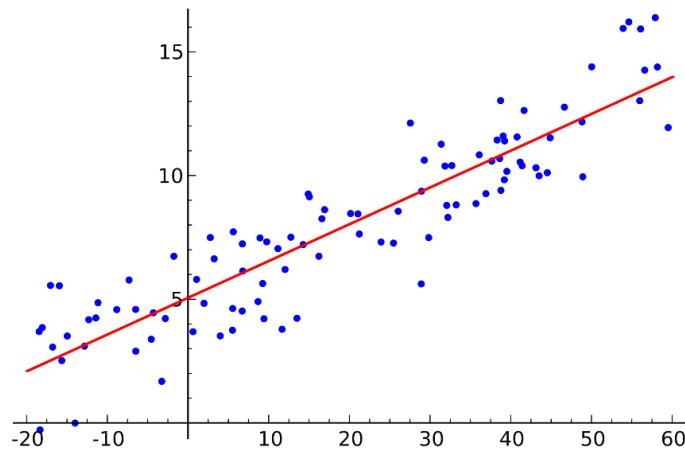


Intuition of hypothesis function

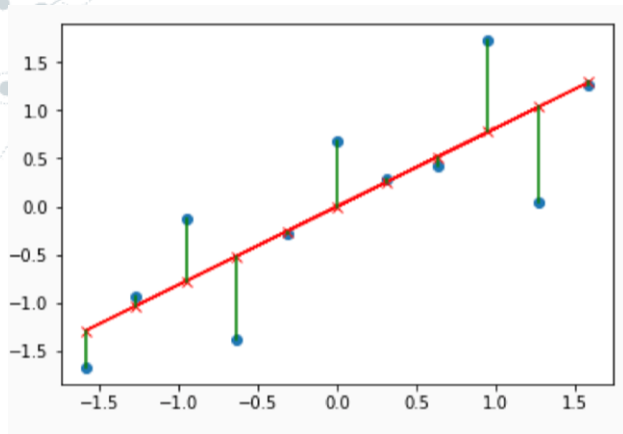
$$f_{\theta}(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Two equivalent questions:

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



Cost function



$$e^{(i)} = \widehat{y}^{(i)} - y^{(i)} = f_{\theta}(x^{(i)}) - y^{(i)}$$

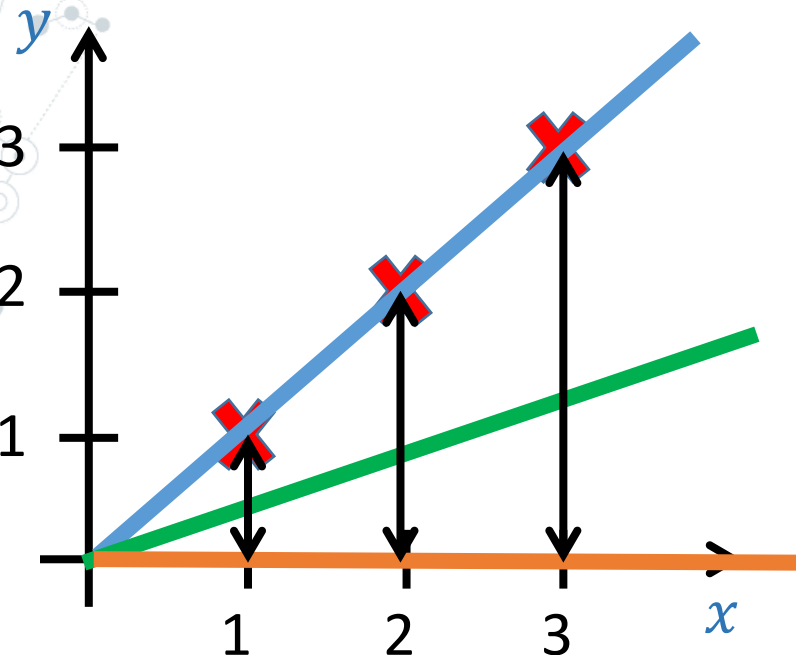
prediction error for i th training example

$$loss(\bar{\theta}) = \frac{1}{2m} \sum_{i=1}^m (f_{\bar{\theta}}(x^{(i)}) - y^{(i)})^2$$

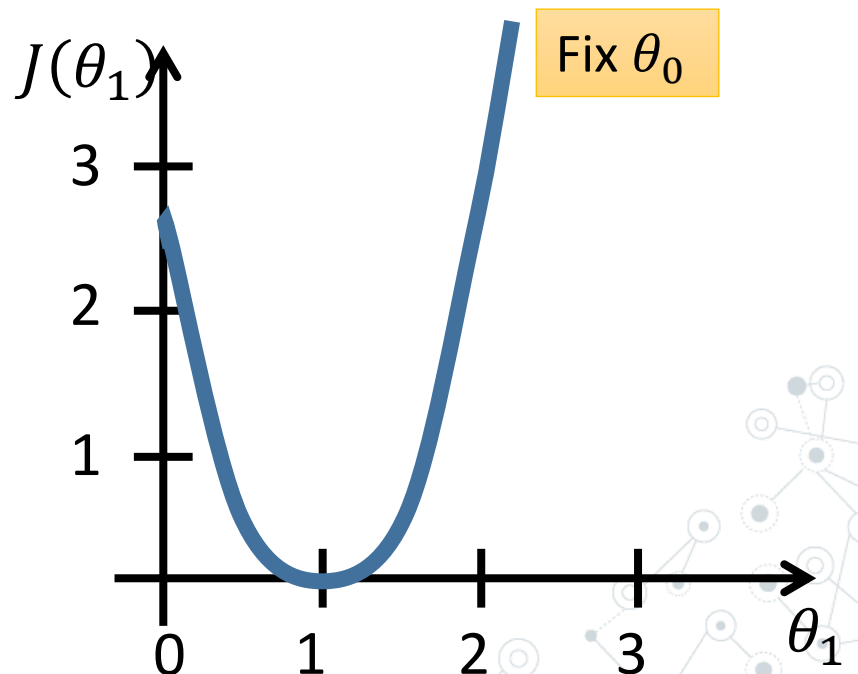
Choose parameters $\bar{\theta}$ so that

$loss(\bar{\theta})$ is minimized

$f_{\theta}(x)$: function of x for fixed θ

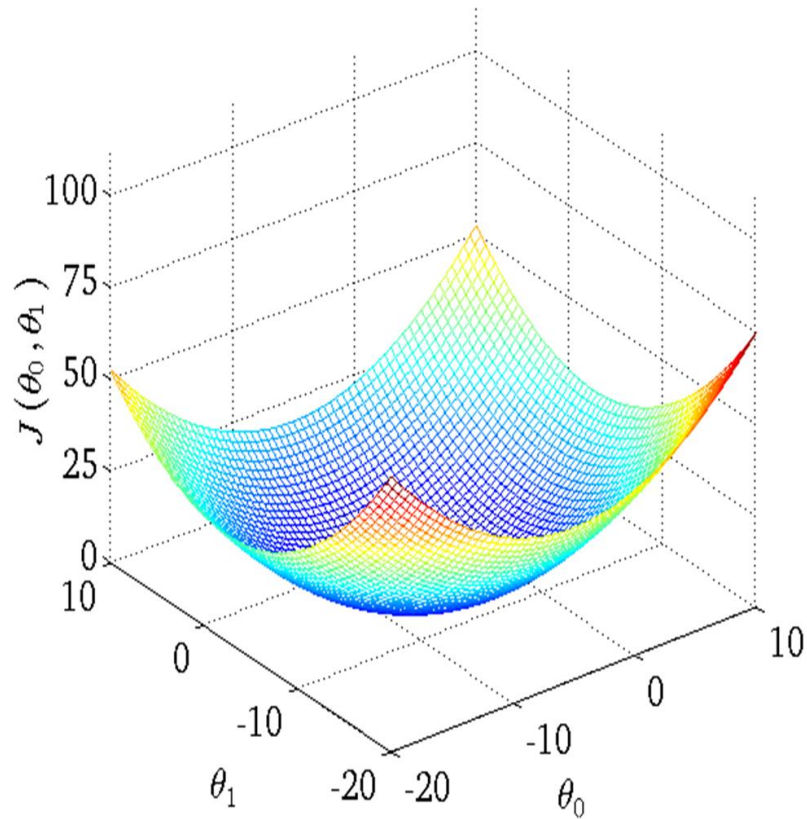


$loss(\theta)$, function of θ_0, θ_1



Cost Function

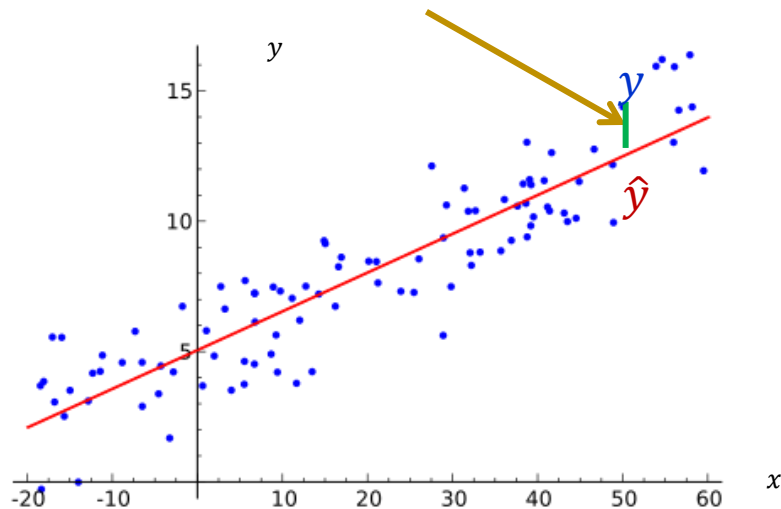
When loss is a function of both θ_0 and θ_1



Linear Regression

$$\hat{y} = \theta_0 + \theta_1 x$$

The loss is the squared loss $L_2(\hat{y}, y) = (\hat{y} - y)^2$



Data (x, y) pairs are the blue points.
The model is the red line.

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

Linear Regression

The total loss across all points is

$$\begin{aligned} L &= \sum_{i=1}^m (\widehat{y^{(i)}} - y^{(i)})^2 \\ &= \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2 \\ \text{loss}(\theta_0, \theta_1) &= \frac{1}{m} \sum_{i=1}^m (f(x^{(i)}; \theta) - y^{(i)})^2 \end{aligned}$$

We want the optimum values of θ_0, θ_1 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



CS60050: Machine Learning

Autumn 2024

Sudeshna Sarkar

Linear Regression Cont.

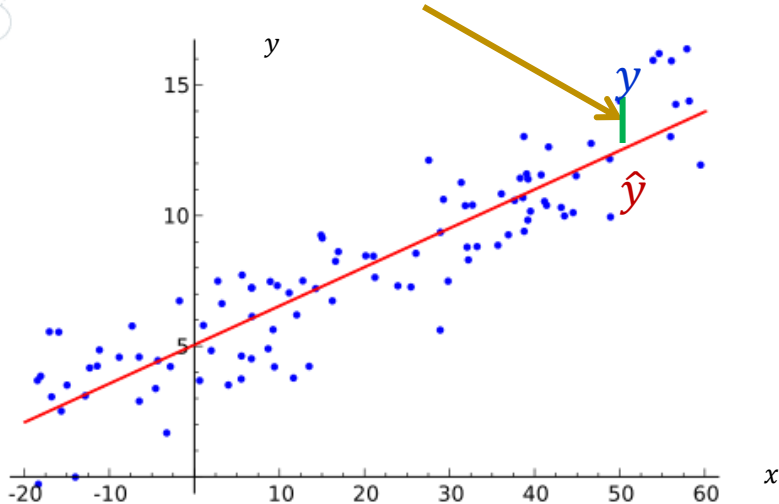
31 July 2024



Linear Regression recap

$$f_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

$$\text{loss}(\bar{\theta}) = \frac{1}{2m} \sum_{i=1}^m (f_{\theta}(x^{(i)}) - y^{(i)})^2$$



In 1 dimension

$$\hat{y} = \theta_0 + \theta_1 x$$

Consider squared loss $L_2(\hat{y}, y) = (\hat{y} - y)^2$

The total loss across all points is

$$L = \sum_{i=1}^m (\widehat{y^{(i)}} - y^{(i)})^2$$

$$= \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

$$\text{loss}(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (f(x^{(i)}; \theta) - y^{(i)})^2$$

x

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

Learning as Optimization Problem

Hypothesis: $f_{\theta}(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function:
$$L = \text{Loss}(\theta) = \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2$$
$$= \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

Goal: $\min_{\theta_0, \theta_1} \text{loss}(\theta_0, \theta_1)$

Linear Regression: Analytic Solution

$$L = \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

Linear Regression: Analytic Solution

We want the loss-minimizing values of θ , so since the loss is differentiable, we set

$$L = \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

$$\frac{dL}{d\theta_0} = 0 \quad \text{and} \quad \frac{dL}{d\theta_1} = 0$$

Linear Regression: Analytic Solution

We want the loss-minimizing values of θ , so since the loss is differentiable, we set

$$L = \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

$$\frac{dL}{d\theta_0} = 0 \quad \text{and} \quad \frac{dL}{d\theta_1} = 0$$

$$\frac{dL}{d\theta_1} = 0 = 2\theta_1 \sum_{i=1}^m (x^{(i)})^2 + 2\theta_0 \sum_{i=1}^m x^{(i)}$$

$$\frac{dL}{d\theta_0} = 0 = 2\theta_1 \sum_{i=1}^m x^{(i)} + 2\theta_0 m - 2 \sum_{i=1}^m y^{(i)}$$

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

Linear Regression: Analytic Solution

We want the loss-minimizing values of θ , so since the loss is differentiable, we set

$$L = \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

$$\frac{dL}{d\theta_0} = 0 \quad \text{and} \quad \frac{dL}{d\theta_1} = 0$$

$$\frac{dL}{d\theta_1} = 0 = 2\theta_1 \sum_{i=1}^m (x^{(i)})^2 + 2\theta_0 \sum_{i=1}^m x^{(i)}$$

$$\frac{dL}{d\theta_0} = 0 = 2\theta_1 \sum_{i=1}^m x^{(i)} + 2\theta_0 m - 2 \sum_{i=1}^m y^{(i)}$$

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

$$\theta_1 = \frac{m \sum_{i=1}^m y^{(i)} x^{(i)} - (\sum_{i=1}^m x^{(i)}) (\sum_{i=1}^m y^{(i)})}{m \sum_{i=1}^m (x^{(i)})^2 - (\sum_{i=1}^m x^{(i)})^2}$$

$$\theta_0 = \frac{1}{m} \left(\sum_{i=1}^m y^{(i)} - \theta_1 \sum_{i=1}^m x^{(i)} \right)$$

Multivariate Linear Regression

$$x \in \mathcal{R}^d$$

$$\hat{y} = f(x; \theta) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_d x_d$$

Define $x_0 = 1$

$$f(x; \theta) = \theta^T \mathbf{x}$$

$$\begin{bmatrix} \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \vdots \\ \hat{y}^{(m)} \end{bmatrix} = \begin{bmatrix} x_0^{(1)} & x_1^{(1)} & x_2^{(1)} & \dots & x_d^{(1)} \\ x_0^{(2)} & x_1^{(2)} & x_2^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_0^{(m)} & x_1^{(m)} & x_2^{(m)} & \dots & x_d^{(m)} \end{bmatrix}$$

$$\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\theta}$$

Cost Function:

$$loss(\boldsymbol{\theta}) = loss(\theta_0, \theta_1, \dots, \theta_d) = \frac{1}{m} (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)})^2$$

Multivariate Linear Regression

$$\text{loss}(\boldsymbol{\theta}) = \frac{1}{m} \sum_i (\boldsymbol{\theta}^T \mathbf{x}^{(i)} - y^{(i)})^2$$

$$= \frac{1}{m} (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})$$

$$J(\boldsymbol{\theta}) = \frac{1}{m} ((\mathbf{X}\boldsymbol{\theta})^T - \mathbf{y}^T) (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})$$

$$= \frac{1}{m} \{(\mathbf{X}\boldsymbol{\theta})^T \mathbf{X}\boldsymbol{\theta} - (\mathbf{X}\boldsymbol{\theta})^T \mathbf{y} - \mathbf{y}^T \mathbf{X}\boldsymbol{\theta} + \mathbf{y}^T \mathbf{y}\}$$

$$\text{loss}(\boldsymbol{\theta}) = \frac{1}{m} \{\boldsymbol{\theta}^T (X^T X) \boldsymbol{\theta} - \boldsymbol{\theta}^T \mathbf{X}^T \mathbf{y} - \mathbf{y}^T \mathbf{X} \boldsymbol{\theta} + \mathbf{y}^T \mathbf{y}\}$$

$$= \frac{1}{m} \{\boldsymbol{\theta}^T (X^T X) \boldsymbol{\theta} - (\mathbf{X}^T \mathbf{y})^T \boldsymbol{\theta} - (\mathbf{X}^T \mathbf{y})^T \boldsymbol{\theta} + \mathbf{y}^T \mathbf{y}\}$$

$$= \frac{1}{m} \{\boldsymbol{\theta}^T (X^T X) \boldsymbol{\theta} - 2(\mathbf{X}^T \mathbf{y})^T \boldsymbol{\theta} + \mathbf{y}^T \mathbf{y}\}$$

$$\|\vec{z}\|_2^2 = \sum_{i=1}^N z_i^2 = \sum_{i=1}^N z_i z_i = \vec{z}^T \vec{z}$$

Multivariate Linear Regression

- Equating the gradient of the cost function to 0,

$$\begin{aligned}\nabla_{\theta} \text{loss}(\boldsymbol{\theta}) &= \frac{1}{m} \{2\mathbf{X}^T \mathbf{X} \boldsymbol{\theta} - 2\mathbf{X}^T \mathbf{y} + 0\} = 0 \\ \mathbf{X}^T \mathbf{X} \boldsymbol{\theta} - \mathbf{X}^T \mathbf{y} &= 0 \\ \boldsymbol{\theta} &= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}\end{aligned}$$

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

$$\frac{\partial \text{loss}(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^m (f_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

Partial derivatives

- Let $y = f(\mathbf{x}) = f(x_1, x_2, \dots, x_n)$ be a multivariate function with n variables
- The mapping is $f: \mathbb{R}^n \rightarrow \mathbb{R}$
- The **partial derivative** of y with respect to its i^{th} parameter x_i is

$$\frac{\partial y}{\partial x_i} = \lim_{h \rightarrow 0} \frac{f(x_1, x_2, \dots, x_i + h, \dots, x_n) - f(x_1, x_2, \dots, x_i, \dots, x_n)}{h}$$

- To calculate $\frac{\partial y}{\partial x_i}$, we can treat $x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n$ as constants and calculate the derivative of y only with respect to x_i
- For notation of partial derivatives, the following are equivalent:

$$\frac{\partial y}{\partial x_i} = \frac{\partial f}{\partial x_i} = \frac{\partial}{\partial x_i} f(\mathbf{x}) = f_{x_i} = f_i = D_i f = D_{x_i} f$$

Multidimensional derivative: Gradient

Gradient vector: The gradient of the multivariate function $f(\mathbf{x})$ with respect to the n -dimensional input vector $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$, is a vector of n partial derivatives

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}, \dots, \frac{\partial f(\mathbf{x})}{\partial x_n} \right]^T$$

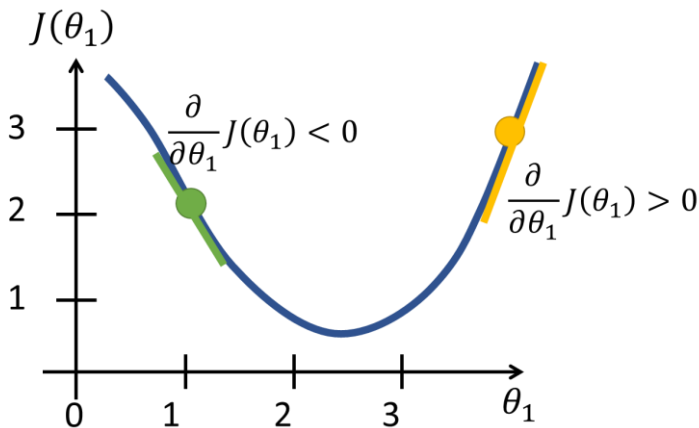
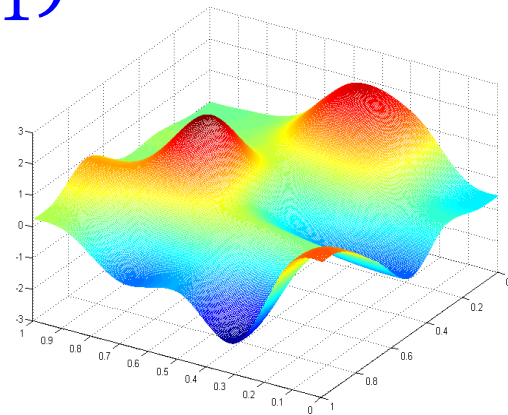
- In ML, the gradient descent algorithm relies on the opposite direction of the gradient of the loss function \mathcal{L} with respect to the model parameters θ ($\nabla_{\theta} \mathcal{L}$) for minimizing the loss function



Minimizing cost function & Gradient Descent

Minimizing function $loss(\theta_0, \theta_1)$

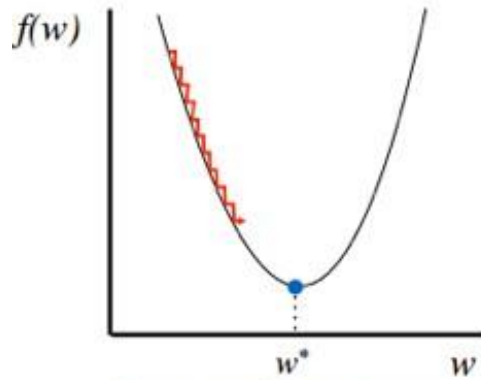
- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $loss(\theta_0, \theta_1)$
- until we end up at a minimum



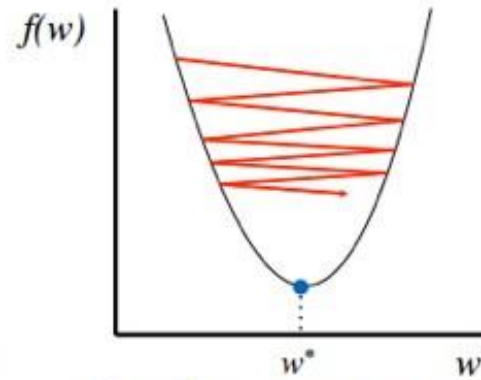
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} loss(\theta_1)$$

Step Size α

- α 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{

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} \text{loss}(\bar{\theta})$$

Equivalently

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

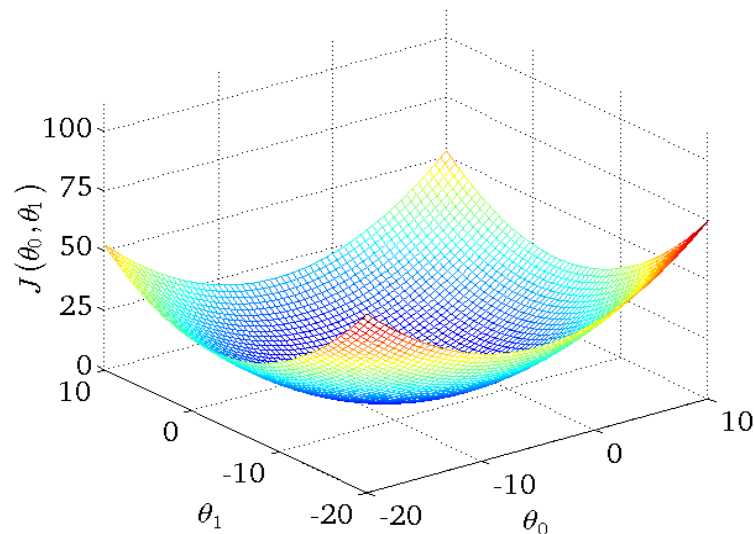
}

$$\text{loss}(\bar{\theta}) = \frac{1}{2m} \sum_{i=1}^m (f_{\theta}(x^{(i)}) - y^{(i)})^2$$

$$\frac{\partial}{\partial \theta_j} \text{loss}(\bar{\theta}) = \frac{1}{m} \sum_{i=1}^m (f_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

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

Repeat until convergence{

m : Number of training examples

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\theta}(x^{(i)}) - y^{(i)})$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (f_{\theta}(x^{(i)}) - y^{(i)}) x^{(i)}$$

}

Gradient Descent

- Batch \rightarrow Mini-batch \rightarrow Stochastic
- Basically, just a difference in how many training points are each time we compute a gradient (and then take a step with that gradient)

(Batch) GD



Effective batch size = m

Mini-batch SGD



Batch size = B , e.g., $B = 10$

SGD



Effective batch size = 1