Outline

- Non-linear model
- Neural network

Linear regression review

$$h_\theta(x) = \theta^T x + b$$

$$J(\theta) = \sum_{i=1}^{n} \left(y^{(i)} - h_\theta(x^{(i)})\right)^2 = \sum_{i=1}^{n} \left(y^{(i)} - \theta^T x^{(i)} - b\right)^2$$

Run GD or SGD

Non-linear model

- Kernel $\quad h_\theta(x) = \theta^T \phi(x)$ linear in θ, but not x

- non-linear in both θ and x

  ex: $h_\theta(x) = \sqrt{\theta_1^3 x_2 + \sqrt{\theta_3} x_4}$

Assume we have a dataset $\{x^{(i)}, y^{(i)}\}_{i=1}^{n}$

$$x^{(i)} \in \mathbb{R}^d, \quad y^{(i)} \in \mathbb{R}$$

$$h_\theta \quad \mathbb{R}^d \to \mathbb{R}$$

- Cost func for example i

$$J^{(i)}(\theta) = \left(y^{(i)} - h_\theta(x^{(i)})\right)^2$$

- Cost func for dataset

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} J^{(i)}(\theta)$$

Note: the minimizer will remain the same w/o $(\frac{1}{n})$

Optimize

$$\min_\theta J(\theta)$$

GD: $\quad \theta := \theta - \alpha \nabla J(\theta)$

SGD:

for $i=1$ to n_{iter}:

  sample j from $\{1, \dots, n\}$

$\theta := \theta - \alpha \nabla J(\theta)$

mini-batch SGD

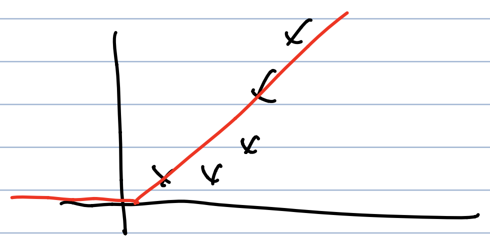
Computing B gradients $\nabla J^{(j_1)}(\theta), \dots \nabla J^{(j_B)}(\theta)$ simultaneously is faster
(batch size)

for $i = 1$ to n_{iter}:
  Sample B examples $j_1, \dots j_B$ without replacement
  $\theta := \theta - \alpha \frac{1}{B} \sum_{k=1}^{B} \nabla J^{(j_k)}(\theta)$

Key Points

① How to define $h_\theta(x)$? Neural Network

② How to compute $\nabla J^{(j)}(\theta)$? Back propagation



In deep learning

$Relu = \max\{t, 0\}$

$h_\theta(x) = \underbrace{relu}_{\text{activation}}(wt + b)$ ← neural network with one neuron

High dimensional input $x \in \mathbb{R}^d$, $y \in \mathbb{R}$

$h_\theta(x) = relu(w^T x + b)$

$\underbrace{w \in \mathbb{R}^d}_{\text{weigh vector}}$ $\quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \in \mathbb{R}^d$, $\quad \underbrace{b \in \mathbb{R}}_{\text{bias}}$

Stacking neurons

Ex: $x \in \mathbb{R}^4$. $\quad x_1, \quad x_2, \quad x_3, \quad x_4$

x_1 ↑ size, x_2 ↑ # of bed, x_3 ↑ zip code

intermediate variables:

family size	a_1
walkable	a_2
school area	a_3

family size $\quad a_1 = relu(\theta_1 x_1 + \theta_2 x_2 + \underbrace{\theta_3}_{\text{bias}})$

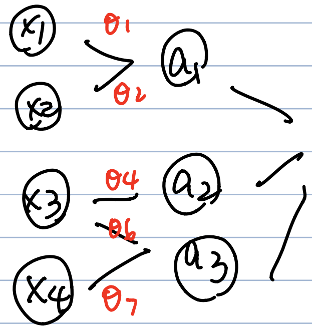
walkable $\quad a_2 = relu(\theta_4 x_3 + \theta_5)$

$a_3 = relu(\theta_6 x_3 + \theta_7 x_4 + \theta_8)$

$h_\theta(x) = \cancel{\text{relu}}(\theta_9 a_1 + \theta_{10} a_2 + \theta_{11} a_3 + \theta_{12}\cancel{)}$

(Generally we don't apply relu at the end)

Diagram



Intermediate values → hidden units

$a_j = \text{relu}(w_j^{[1]T} x + b^{[1]})$ $\forall j = 1 .. m$ - first layer

$h_\theta(x) = w^{[2]T} x + b^{[2]}$ - second layer

Vectorization

$$W^{[1]} = \begin{bmatrix} - & w_1^{[1]T} & - \\ - & w_2^{[1]T} & - \\ & \vdots & \\ - & w_m^{[1]T} & - \end{bmatrix} \in \mathbb{R}^{m \times d}$$

$$W^{[1]} + \begin{bmatrix} b^{[1]} \\ \vdots \\ b^{[m]} \end{bmatrix} = \underbrace{\begin{bmatrix} w_1^{[1]} x + b^{[1]} \\ \vdots \\ w_m^{[1]} x + b^{[m]} \end{bmatrix}}_{\mathbb{R}^m} \triangleq \begin{bmatrix} z_1 \\ \vdots \\ z_m \end{bmatrix}$$

$$z = W^{[1]} x + b^{[1]} \in \mathbb{R}^m$$

$$a = \begin{bmatrix} \text{relu}(z_1) \\ \vdots \\ \text{relu}(z_m) \end{bmatrix} \triangleq \text{relu}(z)$$

$$W^{[2]} = [\, w^{[2]T} \,] \in \mathbb{R}^{1 \times m}, \quad b^{[2]} \in \mathbb{R}$$

$$h_\theta(x) = W^{[2]} a + b$$

Stack more layers

$$a^{[1]} = \mathrm{relu}(w^{[1]}x + b^{[1]})$$

$$a^{[2]} = \mathrm{relu}(w^{[2]}x + b^{[2]})$$

$$\vdots$$

$$a^{[r-1]} = \mathrm{relu}(w^{[r-1]}x + b^{[r-1]})$$

$$h_\theta = W^{[r]}a^{[k-1]} + b^{[r]}$$

$\dim(a_k) = m_k$ $\quad x \in \mathbb{R}^d, \quad w^{[1]} \in \mathbb{R}^{m_1 \times d}$

$W^{[1]}x \in \mathbb{R}^{m_1} \qquad b^{[1]} \in \mathbb{R}^{m_1}$

$W^{[2]} \in \mathbb{R}^{m_2 \times m_1}$

$W^{[k]} \in \mathbb{R}^{m_k \times m_{k-1}} \qquad b^{[k]} \in \mathbb{R}^{m_k}$

$$\text{width} = \max\{m_1, \ldots m_{r-1}\}$$