Deep Learning

Supervised learning w/ non-linear models

before Kernel method $h_\theta(x) = \theta^T \phi(x)$ non-linear in x, linear θ

eg. $h_\theta(x) = \sqrt{\theta_1^3 + \theta_2 x_3 + \sqrt{\theta_4 x_7}}$

dataset $\{(x^{(i)}, y^{(i)})\}_{i=1}^n$ $x^{(i)} \in \mathbb{R}^d$, $y^{(i)} \in \mathbb{R}$

$h_\theta(x) : \mathbb{R}^d \to \mathbb{R}$

cost/loss f^n

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

cost f^n for entire dataset

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

optimization objective

$\min_\theta J(\theta)$

gradient descent

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

Stochastic Gradient Descent (SGD)

for $i = 1$ to n_{iter}

Sample j from $\{1 \ldots n\}$ randomly

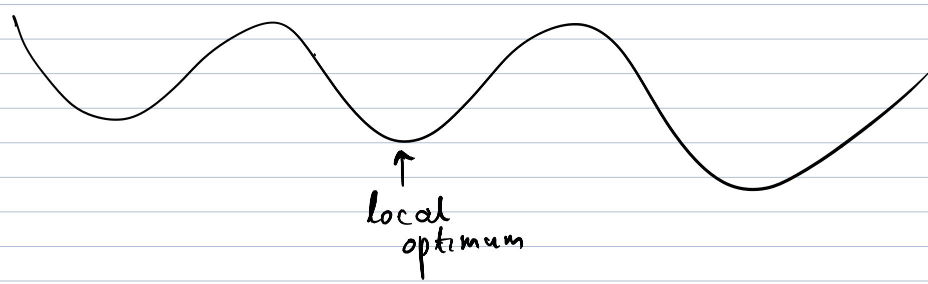
$\theta := \theta - \alpha \nabla J^{(j)}(\theta)$

minibatch SGD

Idea: Computing B gradients $\nabla J^{(j_1)}(\theta) \ldots \nabla J^{(j_B)}(\theta)$ together is faster than sequential computation for $i = 1$ to n_{iter}

Sample B examples $\{j_1 \dots j_B\}$ from $\{1 \dots n\}$

$$\theta := \theta - \frac{\alpha}{B} \sum_{k=1}^{B} \nabla J^{(j_k)}(\theta)$$



remaining questions

① How to define $h_\theta(x)$?

② how to compute gradient

– Logistic Regression

– Neural Networks

– computational power

– data available

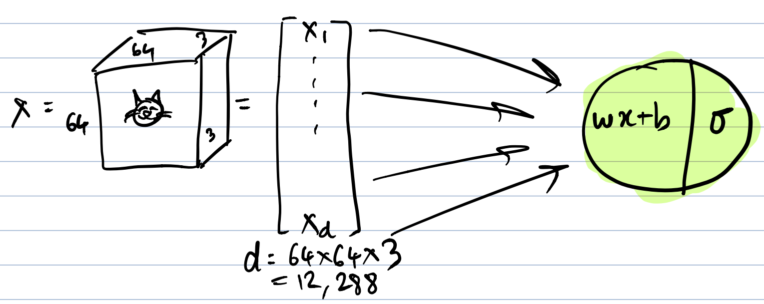
– algorithms

① Logistic Regression

goal Find cats in images

1 → presence of cat

0 → absence of cat



$$\hat{y} = \sigma(\theta^T x)$$

$$= \sigma(wx + b)$$

w: 1×12288 ; x: 12288×1

(i) initialize $\overset{\text{weights}}{\uparrow} w$, $\overset{\text{bias}}{\downarrow} b$

(ii) Find optimal w, b

(iii) Use $\hat{y} = \sigma(wx+b)$ to predict

$$\mathcal{L} = -\left[y \log \hat{y} + (1-y)\log(1-\hat{y})\right]$$

$$w = w - \alpha \frac{\partial \mathcal{L}}{\partial w}$$

$$b = b - \alpha \frac{\partial \mathcal{L}}{\partial b}$$ ← Note: fixed typo in the live notes

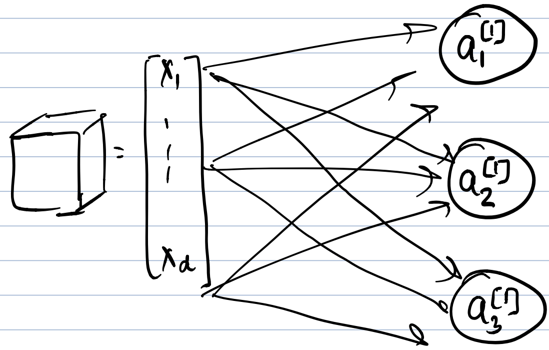
parameters $= d+1$

$12288 + 1$

neuron = linear + activation

model = architecture + parameters

Goal 2.0: Find cat/lion/iguana in images



$$\hat{y}_1 = a_1^{[1]} = \sigma(w_1^{[1]}x + b_1^{[1]})$$

$$\hat{y}_2 = a_2^{[1]} = \sigma(w_2^{[1]}x + b_2^{[1]})$$

$$\hat{y}_3 = a_3^{[1]} = \sigma(w_3^{[1]}x + b_3^{[1]})$$

square brackets $[1] \equiv$ layer

subscripts $\equiv$ identify neuron within layer

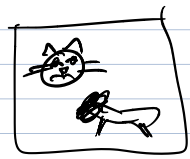
$a_2^{[1]}$ ← 1st layer

← 2nd neuron in 1st layer

param = 3(d+1)

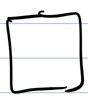
images + labels

 $= \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ — cat — lion — iguana

 $= \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$ cat lion iguana

Goal 3.0 :

Add constraint : unique animal in image

 $= \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}$

$z_1^{[1]}$

$z_2^{[1]}$

$z_3^{[1]}$

$$\hat{y}_1 = e^{z_1^{[1]}} \Big/ \sum_{k=1}^{3} e^{z_k^{[1]}}$$

$$\hat{y}_2 = e^{z_2^{[1]}} \Big/ \sum_{k=1}^{3} e^{z_k^{[1]}}$$

$$\hat{y}_3 = e^{z_3^{[1]}} \Big/ \sum_{k=1}^{3} e^{z_k^{[1]}}$$

Note: Fixed typo in live notes

$$\hat{y}_1 = \sigma(\underbrace{w_1^{[1]} x + b_1^{[1]}}_{z_1^{[1]}})$$

Softmax multiclass network

$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ cat $\quad \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ lion $\quad \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ iguana

parameters = 3(d+1)

Loss function

$$\mathcal{L}_{3N} = -\sum_{k=1}^{3} \left[y_k \log \hat{y}_k + (1-y_k) \log (1-\hat{y}_k) \right]$$

Binary Cross Entropy Loss Function

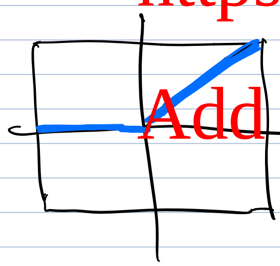
$$\mathcal{L}_{CE} = -\sum_{k=1}^{3} y_k \log \hat{y}_k$$

Q^n: Predict age of cat instead of presence of cat?

Options

① Several neurons to predict different ages

② Change activation fn



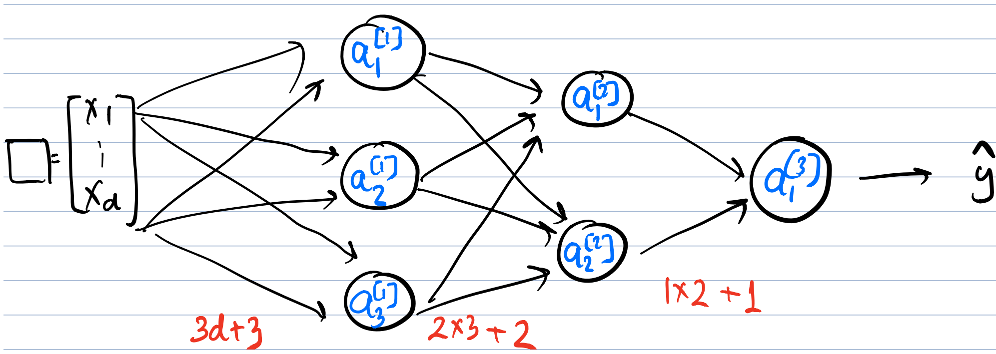
$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \le 0 \end{cases}$$

ReLU

Rectified Linear Unit

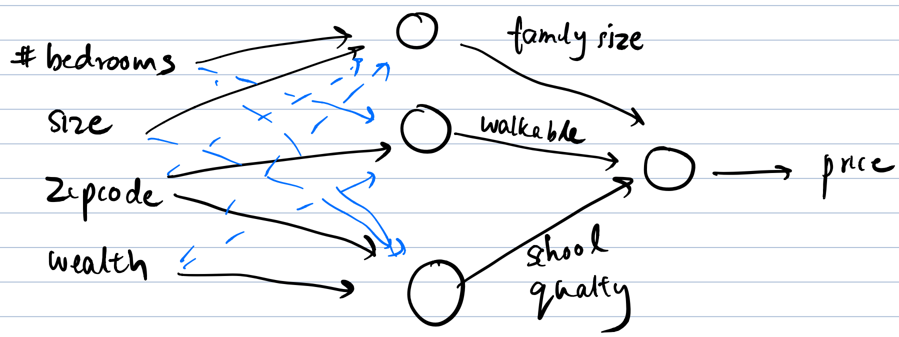
Modified Loss fn $\qquad \|\hat{y} - y\|_1 \qquad \|\hat{y} - y\|_2^2$

② Neural Networks



input layer, output layer
hidden layer

House price prediction



Propagation equations

$3\times1 - z^{[1]} = \underset{3\times d}{W^{[1]}} \underset{d\times 1}{x} + \underset{3\times1}{b^{[1]}}$ $\qquad Z^{[1]} = \underset{3\times n}{} \underset{3\times d}{W^{[1]}} \underset{d\times n}{X} + b^{[1]}$

Broadcasting

$3\times1 - a^{[1]} = \sigma(z^{[1]})$

$b^{[1]} = \underbrace{\begin{bmatrix} b^{(1)} & \cdots & b^{[1]} \end{bmatrix}}_{n \text{ copies}}$

$2\times1 - z^{[2]} = \underset{2\times3}{W^{[2]}} \underset{3\times1}{a^{[1]}} + \underset{2\times1}{b^{[2]}}$

$2\times1 - a^{[2]} = \sigma(z^{[2]})$

$1\times1 - z^{[3]} = \underset{1\times2}{W^{[3]}} \underset{2\times1}{a^{[2]}} + \underset{1\times1}{b^{[3]}}$

$1\times1 - a^{[3]} = \sigma(z^{[3]})$

$X = \begin{bmatrix} x^{(1)} & \cdots & x^{(n)} \end{bmatrix}$

[ ] — layer
( ) — id of example
capital — batch of examples

Optimize $w^{[1]}$ $w^{[2]}$ $w^{[3]}$ $b^{[1]}$ $b^{[2]}$ $b^{[3]}$

Define Loss f^n