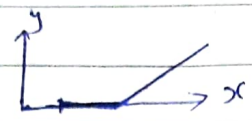
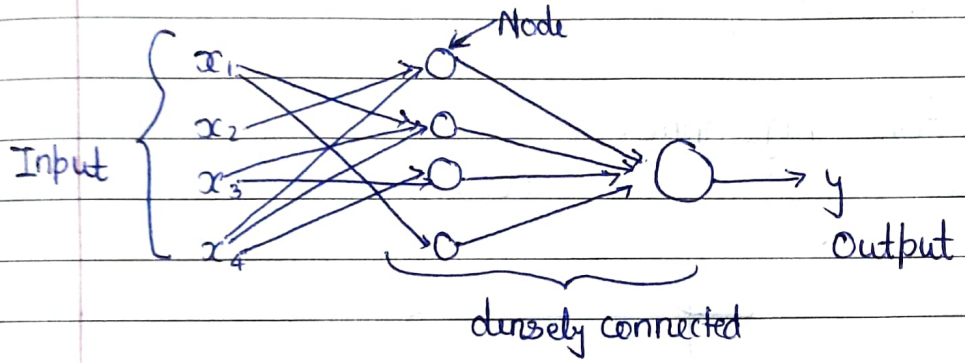
Nueral Network and Deep Learning



ReLU function
└ Rectified Linear Unit

Neuron is a single function. A grp of neurons from a network.



Supervised learning

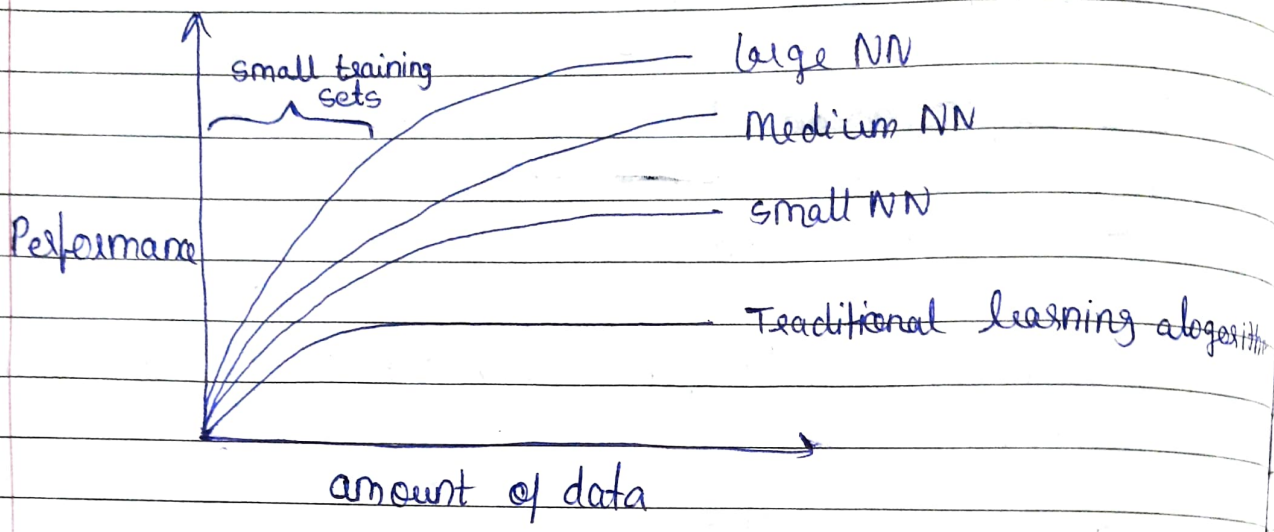
	Input (x)	Output (y)	Application
Ex.	Ad, user info	click on ad? (0/1)	Online advertising
	English	Other lang	Machine translating
CNN {	Image	Object (1, ..., 1000)	Photo tagging

You should figure out what is X and what is Y.

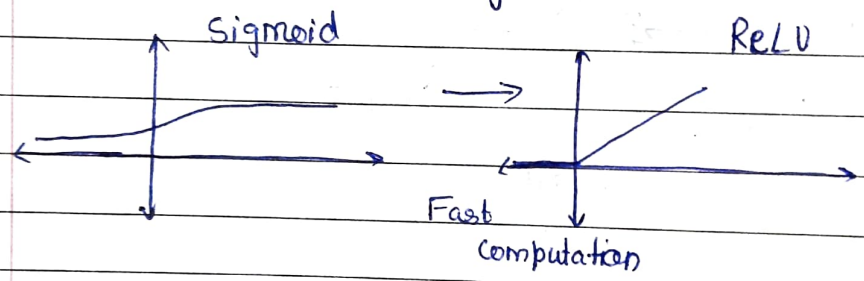
Recurrent NN — One dimensional structured data.

Structured data	Unstructured date
survey data, tabulated	Audio, image, texts
makes sense to computers	harder to make sense

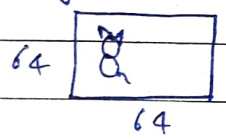
Why deep learning is getting popular recently?



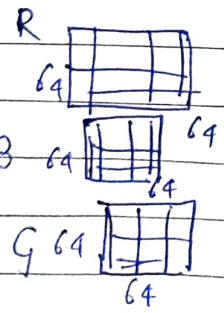
Data, Computation, Algorithms.



Logistic regression is an algorithm for binary classification



→ 1 (cat) vs 0 (Non-cat)



feature vector

$$X = \begin{bmatrix} \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{bmatrix}$$

$64 \times 64 \times 3 = 12288$

$n = n_x = 12288$

dimension

$(x, y) \quad x \in \mathbb{R}^{n_x}, \quad y \in \{0, 1\}$

m training ex: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots\}$

{training set}

$M = M_{train}$ $\qquad$ $m = m_{test}$

$$X \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix} \updownarrow n_x$$

$X \in \mathbb{R}^{n_x \times m}$ $\qquad$ $X_{shap} = (n_x, m)$

$Y = [y^{(1)}, y^{(2)} \cdots y^{(m)}]$

$Y \in \mathbb{R}^{1 \times m}$

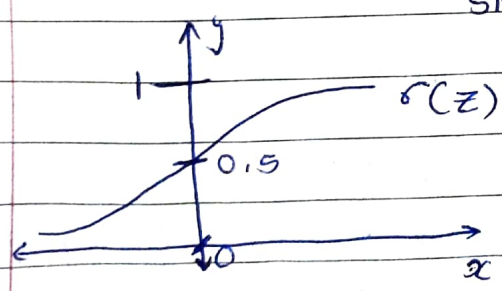
$Y_{shape} = (1, m)$

$\hat{y} = P(y=1|x) \quad 0 \le \hat{y} \le 1$

estimate/ prediction of y

Parameters: $w \in \mathbb{R}^{n_x}$, $b \in \mathbb{R}$ – Real number.

Output $\Rightarrow \hat{y}^{(i)} = \underbrace{\sigma(\underbrace{w^T x^{(i)} + b}_{z})}_{\text{sigmoid}} = y^{(i)}$ – close to the actual value



$\sigma(z) = \dfrac{1}{1+e^{-z}}$

If z large,
$\sigma(z) \approx \dfrac{1}{1+0} = 1$

If z is small,
$\sigma(z) \approx 0$

w & b are seperate parameters

Loss function

$$\mathcal{L}(\hat{y}, y) = -(y \log \hat{y} + (1-y) \log(1-\hat{y}))$$

How close $\hat{y}$ is to y

If $y = 1$

$\mathcal{L}(\hat{y}, y) = -(\log \hat{y}) \leftarrow$ want this to be large

$\hat{y}$ should be large

If $y = 0$

$\mathcal{L}(\hat{y}, y) = -\log(1-\hat{y}) \leftarrow$ want $\log(1-\hat{y})$ to be large

$\hat{y}$ should be small)

Cost func^n

$$: J(w, b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$$

$$= -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log \hat{y}^{(i)} + (1-y^{(i)}) \log(1-\hat{y}^{(i)}) \right]$$

How well the NN is doing on a single training example

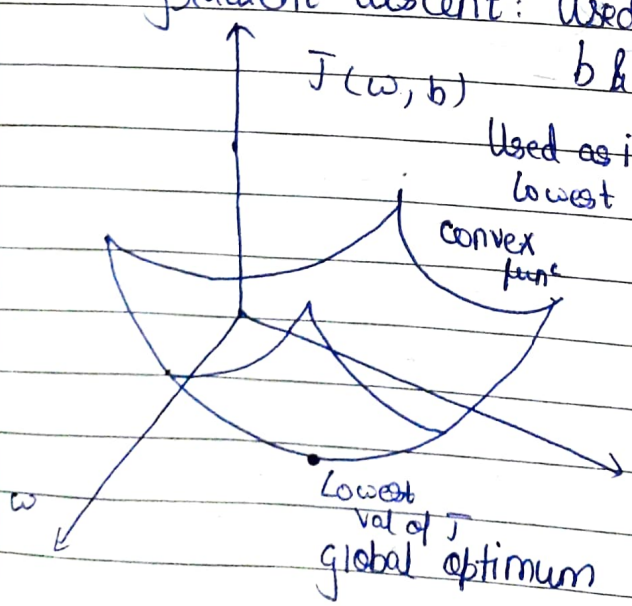
$\mathcal{L}(\,,\,)$ applied to a single training example.

Cost$(\,,\,)$ - cost of the parameters

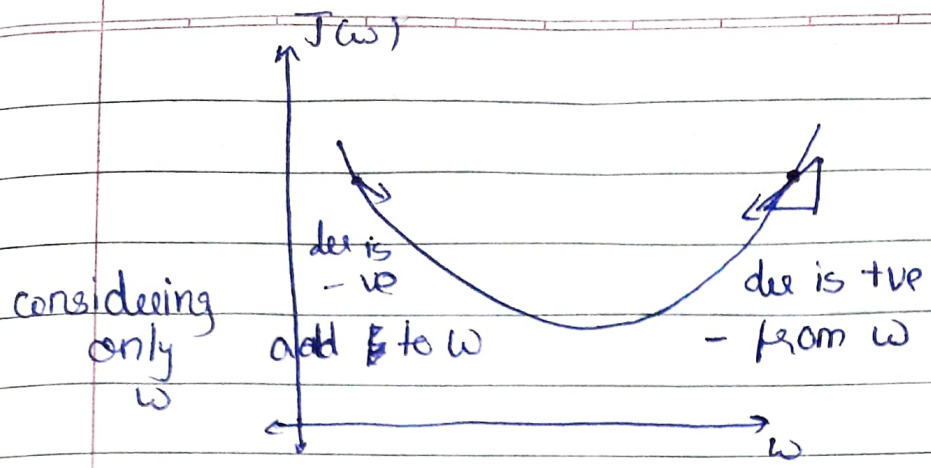
Find w, b and try to minimize J

How well parameters w & b are doing on the entire training set.

Gradient descent: Used to choose the val of b & w



We can initialize parameters by any val as func is convex. We reach the same point



$$\omega := \omega - \alpha \frac{dJ(\omega)}{d\omega}$$

learning rate, how big the next val will be

$$b : b - \alpha \frac{\partial J(\omega, b)}{\partial b}$$

$$\omega : \omega - \alpha \frac{\partial J(\omega, b)}{\partial \omega}$$

Computation Graph

Ex - $J(a,b,c) = 3(a + \underbrace{\underbrace{bc}_{u}}_{v})$ $\quad J$

$u = bc, \quad v = a + u, \quad J = 3v$

$$\begin{tikzcd}
a \arrow[rr] & & \boxed{v = a+u} \arrow[r] & \boxed{J = 3v} \\
b \arrow[r] & \boxed{u = bc} \arrow[ur] & & \\
c \arrow[ur] & & &
\end{tikzcd}$$

Forward path - output
Backward path - Gradients, derivatives

$$\frac{dJ}{dv} = 3 \qquad \frac{dJ}{da} = \frac{dJ}{dv} \cdot \frac{dv}{da}$$