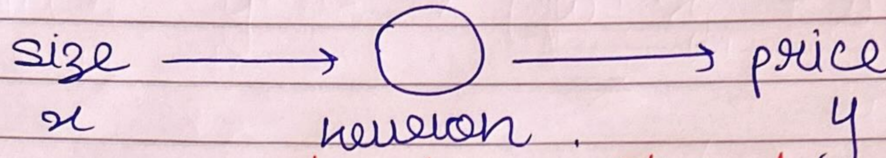
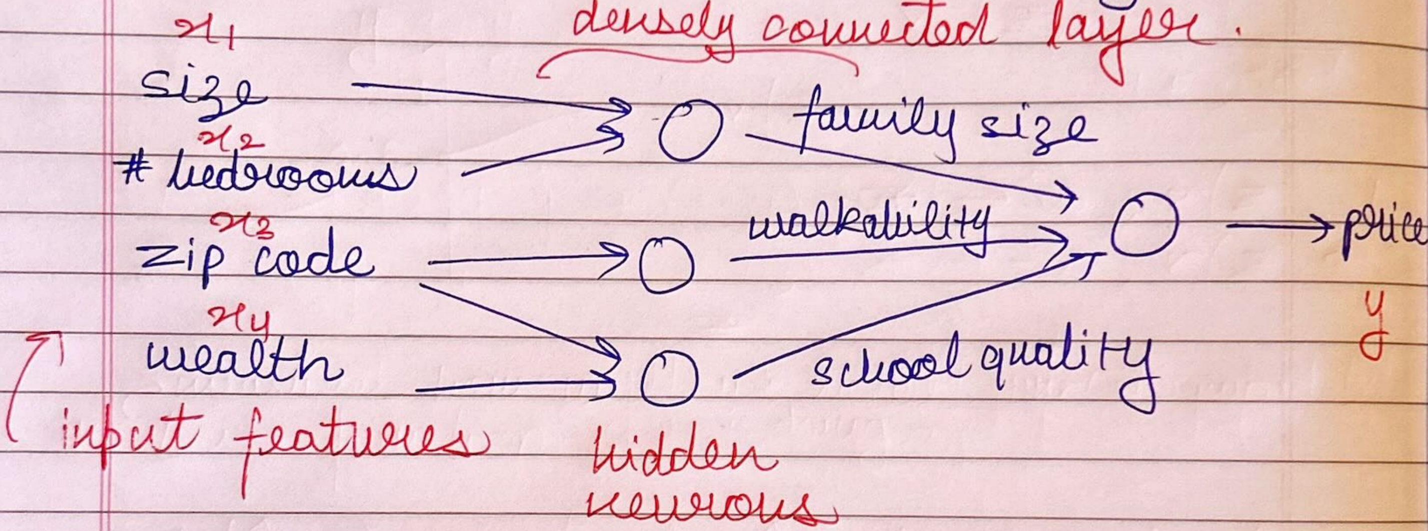
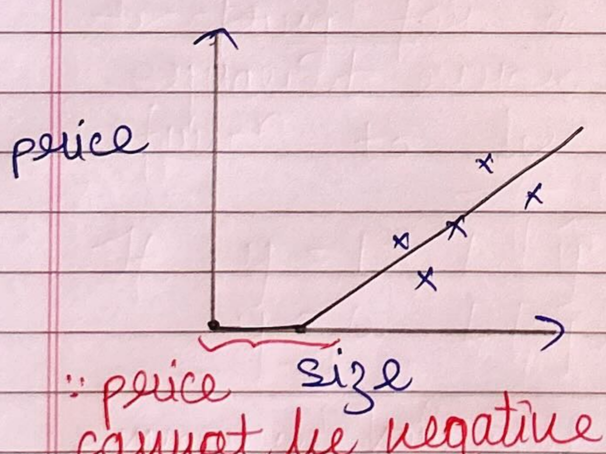
19/7/24

DEEP LEARNING

* What is a neural network?
Stacking together many neurons gives us a neural network.







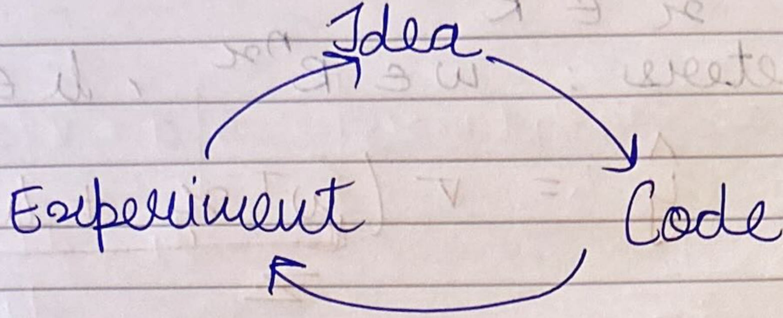
Relu function
(Rectified Linear Unit)

∵ price cannot be negative

* Common Neural Networks

CNN	RNN	Hybrid/Custom
→ images	→ sequential data	→ advanced models.

Structured Data	Unstructed Data
Proper data set, every feature is defined	Images, audio, random texts.



as building an effective NN is an iterative process.

* BINARY CLASSIFICATION:
input image : 64 x 64.

feature vector : matrix $X = \begin{bmatrix} 255 \\ 231 \\ \vdots \\ 255 \\ 231 \\ \vdots \end{bmatrix} = 64 \times 64 \times 3$ (∵ RGB)

$n_x = 12288$ (here)

$x \longrightarrow y$
image of cats 0/1

no. of
m = training examples

Basic Notations

★ $X \in R^{n_x \times m}$ ($n_x \times m$ dimensional matrix)

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

$\longleftarrow m \longrightarrow$

★ $Y \in R^{1 \times m} = [y^{(1)} \; y^{(2)} \; \ldots \ldots \; y^{(m)}]$

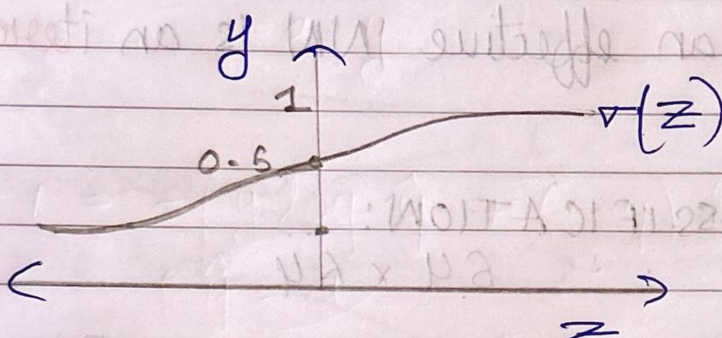
* LOGISTIC REGRESSION:

$$\hat{y} = P(y = 1 \mid x)$$

~~$x \in$~~ $x \in R^{n_x}$

Parameters : $w \in R^{n_x}$, $b \in R$.

Output : $\hat{y} = \sigma(\underbrace{w^T x + b}_{z})$



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

$z \to \infty$; $\sigma(z) = 1$

$z \to -\infty$; $\sigma(z) \approx 0$.

→ measures how well our model func^n (sigmoid here) is doing

* Loss function :

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

→ opting for a sigmoid function over squared error here as we want the loss func^n to be convex.

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

$y = 0$ $L(\hat{y}, y) = -\log(1-\hat{y})$

→ measures how well parameters w, b are doing

* Cost function :

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

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

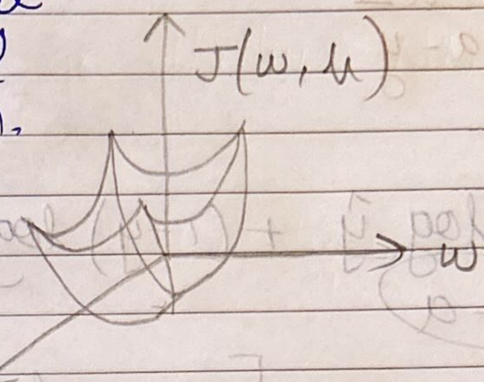
* GRADIENT DESCENT :

→ want to find w, b that minimizes $J(w, b)$

$$w = w - \alpha \frac{\partial J(w, b)}{\partial w}$$

$$b = b - \alpha \frac{\partial J(w, b)}{\partial b}$$

With any initial value, you will reach the min. (path maybe diff.).



Why gradient descent?
→ it is a convex func^n. Thus, has one minima only (global)

* COMPUTATION GRAPH :

$J(a, b, c) = 3(a + bc)$

$$\begin{array}{l} \overset{da = 3}{a} \longrightarrow \\ b \longrightarrow \boxed{u = bc} \longrightarrow \boxed{V = a + u} \longrightarrow \boxed{J = 3V} \\ c \longrightarrow \end{array}$$

$du = 3$ ⟵ $dv = 3$ ⟵ derivative

We observe, through a left to right pass we computed the value of J.

To compute derivatives, we move from right to left, i.e., in technical terms use backpropagation to calculate derivative.

★ dvar in code ≡ J (here).

in code, $\frac{dJ}{da} = da = \underline{3}$ (here)

* Gradient Descent for Logistic Regression (using computation graph)

$$\left.\begin{matrix} x_1 \\ w_1 \\ x_2 \\ w_2 \\ b \end{matrix}\right\} \longrightarrow \boxed{z = w_1x_1 + w_2x_2 + b} \rightleftarrows \boxed{a = \sigma(z)} \rightleftarrows \boxed{L(a,y)}$$

$dz = \dfrac{dL}{dz}$

$= a - y$

$da = \dfrac{dL(a,y)}{da}$

$= \dfrac{-y}{a} + \dfrac{1-y}{1-a}$

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

$\hat{y} = a$

$$da = \frac{\partial L}{\partial a} = -\left[\frac{y}{a} + \frac{-(1-y)}{(1-a)}\right] = -\frac{y}{a} + \frac{(1-y)}{(1-a)}.$$

$$dz = \frac{dL}{dz} = \frac{\partial L}{\partial a} \times \frac{\partial a}{\partial z}$$

$$= \left(\frac{-y}{a} + \frac{1-y}{(1-a)}\right) \frac{\partial}{\partial z}\sigma(z)$$

$$= \left(\frac{-y}{a} + \frac{1-y}{(1-a)}\right) \times \underbrace{\frac{\partial}{\partial z}\frac{1}{1+e^{-z}}}$$

$$\frac{\partial a}{\partial z} = \frac{+1}{(1+e^{-z})^2}\left(e^{-z}\right)$$

$$a = \frac{1}{1+e^{-z}} \Rightarrow e^{-z} = \frac{1}{a} - 1$$

$$\frac{\partial a}{\partial z} = \cancel{a^2}\frac{(1-a)}{\cancel{a}} = a(1-a).$$

$$\therefore dz = \frac{dL}{dz} = \frac{\left(-\cancel{\frac{y}{a}} \cancel{+ay} + a \cancel{-ay}\right)}{\cancel{a(1-a)}} \cancel{a(1-a)}$$

$$dz = \underline{\underline{a-y}}$$

* Logistic Regression on m examples
  - using for loop

for $i: 1 \rightarrow m$

$z^{(i)} =$

$a^{(i)} =$

$J_t =$

$dz = a^{(i)} - y^{(i)}$

$dw_1 \mathrel{+}=$] considering only w_1 & w_2

$dw_2 \mathrel{+}=$

$db \mathrel{+}=$

$J /= m$

$dw_1 /= m$; $dw_2 /= m$; $db /= m$

~~Usi~~ (greater datasets are involved)

As we advance in DL, for loops can be time-consuming, thus we use vectorization.

* VECTORIZATION

$$z = \underbrace{np.dot(w, x)}_{w^T x} + b$$