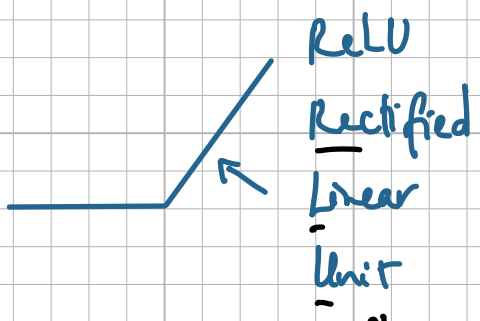
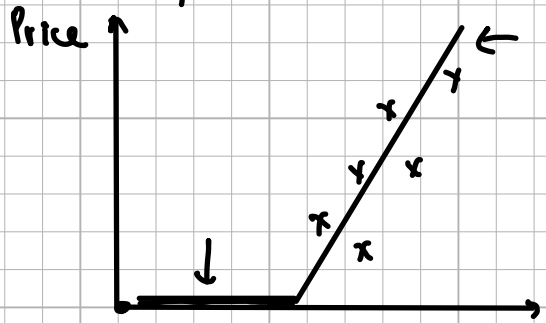
Coursera - Deep Learning

Week 1: Supervised learning

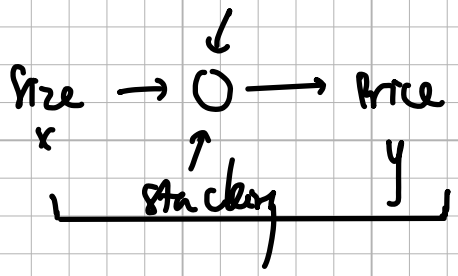
+) Neural network

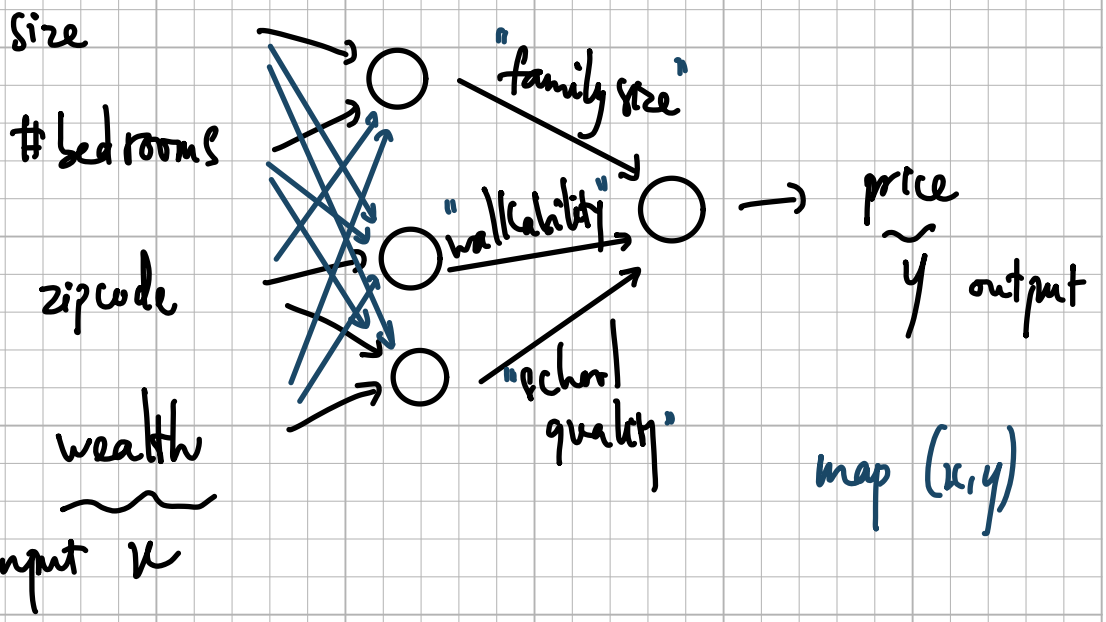


Housing Price Prediction



"neuron"





Supervised Learning

Input (x)	Output (y)	Application
House features	Price	Real estate
Ad, user info	Click on Ad (0/1)	Online Advertising
Image	object (1 ... 1000)	CNN Photo Tagging

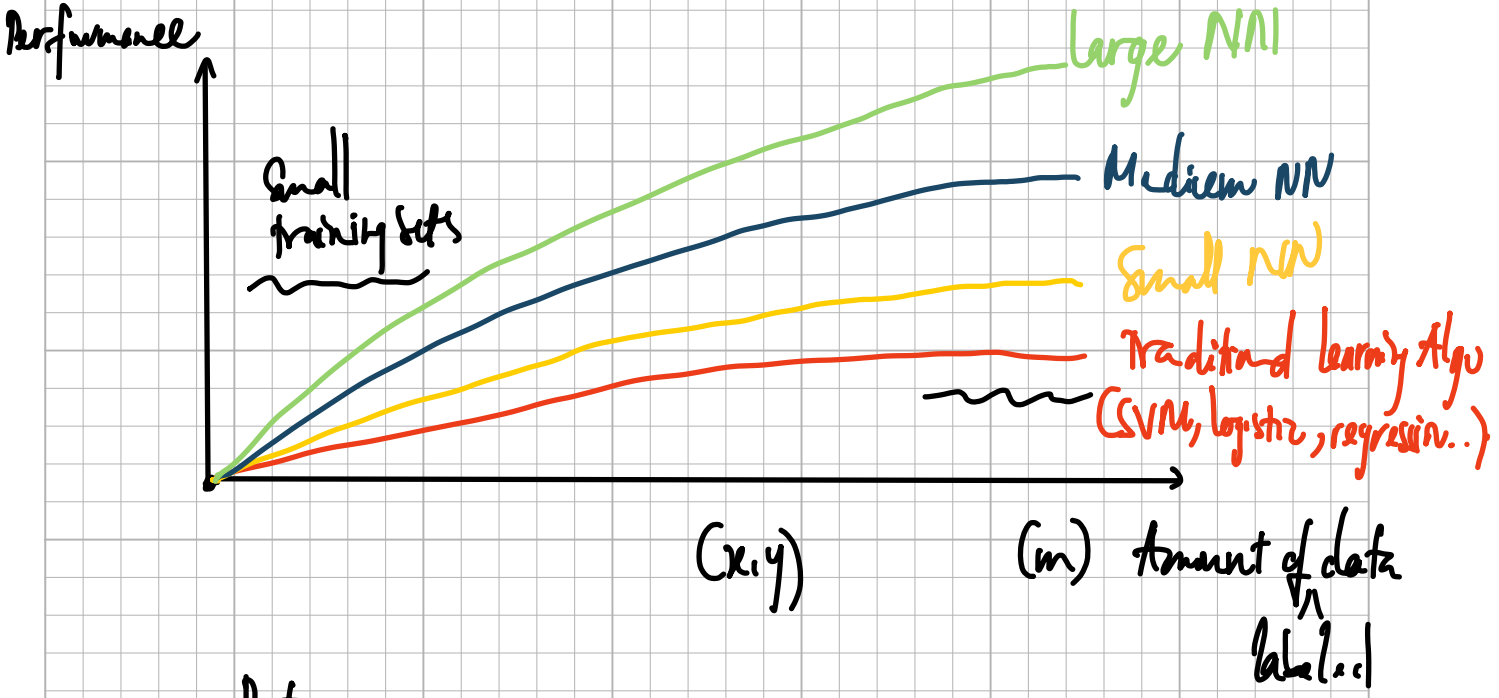
Standard NN

Audio	Text transcript	RNN / Speech recognition
English	Chinese	Machine translation
Image, Radar info	Position of other cars	Custom / Hybrid / Autonomous driving

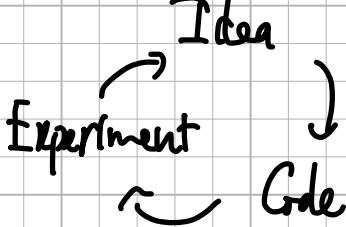
Structured Data
↳ Data like well-defined

Unstructured Data
↳ Audio, Image
Text

- Main drivers:



- Data
- Computation
- Algorithms
  Sigmoid → ReLU



deep unstructured Boltzmann machine

+) Basic NN Programming

Week 2 : Binary Classification

Input $x = \begin{bmatrix} \\ \vdots \\ \\ \end{bmatrix}$ ↕ red ↕ Green ↕ Blue $\quad n_x = 64 \times 64 \times 3$ $\quad x \to y \ (1/0)$

Notation

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

m training examples $\{(x^{(1)}, y^{(1)}), \dots (x^{(m)}, y^{(m)})\}$

$m = m_{train}$ $\quad m_{test}$ = # test examples

$$X = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix} \quad \leftarrow m \rightarrow$$

$X \in \mathbb{R}^{n_x \times m}$

$X.shape = (n_x, m)$

$$Y = \begin{bmatrix} y^{(1)} & y^{(2)} & \cdots & y^{(m)} \end{bmatrix}$$

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

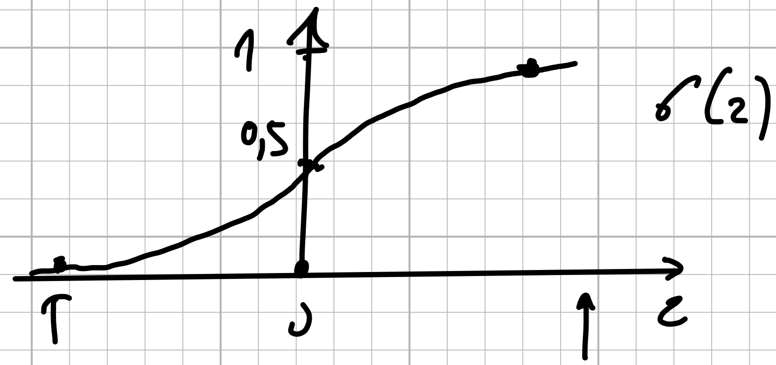
$Y.shape = (1, m)$

+) Logistic Regression

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

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

Output $\hat{y} = \sigma(\underbrace{w^T x + b}_{z})$ $\sigma(z) = \frac{1}{1+e^{-z}}$



- if x large
  $\sigma(z) \approx 1$
- if x large neg
  $\sigma(z) \approx 0$

$x_0 = 1,\ x \in \mathbb{R}^{n_x+1}$

(Not used in course) $\hat{y} = \sigma(\theta^T x)$

$$\theta = \begin{bmatrix} \theta_0 \\ \vdots \\ \theta_{nx} \end{bmatrix} \begin{matrix} \}\ b \\ \\ \}\ w \end{matrix}$$

+) Cost function

$\hat{y}^{(i)} = \sigma(\underbrace{w^T x^{(i)} + b}_{z^{(i)}})$ $x^{(i)}, y^{(i)}, z^{(i)}$ i^{th} example

→ loss (error) function:

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

→ Cost function

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

ab + ac − b − c

a(b + c)

(a − 1)

+) Gradient Descent

Repeat {

$$w := w - \alpha \frac{dJ(w,b)}{dw}$$ (learning rate)

$$b := b - \alpha \frac{d(J,w,b)}{db}$$

}

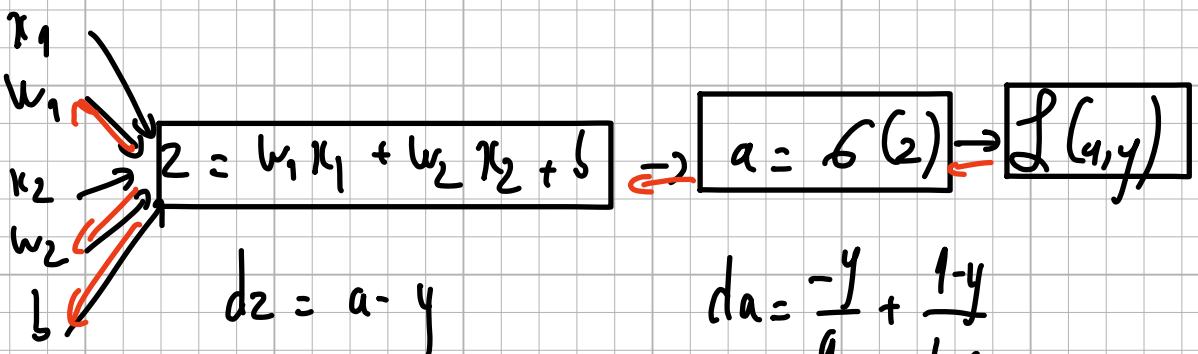
e) Computation graph

forward propagation step → Compute output

backward propagation step → Compute gradients

$$\frac{dJ}{dvar} = "dvar"$$

f) Logistic regression derivatives



$dz = a - y$

$da = \frac{-y}{a} + \frac{1-y}{1-a}$

$dw_1 = x_1 dz$ $dw_2 = x_2 dz$ $db = dz$

ow m examples

→ Vectorization

+) Vectorization

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

GPU } SIMD - single instruction
CPU } multiple data

(n_x, m)

$Z = w^T X + b$ (broadcasting)

$= np.dot(w^T, X) + b$

$A = \sigma(Z)$ $\quad dw = \frac{1}{m} X\, dZ^T$

$db = \frac{1}{m} np.sum(dZ)$ $\quad dZ = A - Y$

→ $cal = A.sum(axis = 0)$

Broadcasting $(1, n) \rightsquigarrow (m, n)$

$(m, 1) \rightsquigarrow (m, n)$

Don't use $(n,)$ "rank 1 array"

.reshape

Week 3: Shallow Neural Network

$\hat{y} = a$ +) Neural Network Representation

layer 1

$a_1^{[1]}$ ...

x

n-layer NN (not count input layer)

$W^{[1]}$

$b^{[1]}$

x, w, b $\rightarrow z = w^T x + b \rightarrow a = \sigma(z) \rightarrow \mathcal{L}(a, y)$

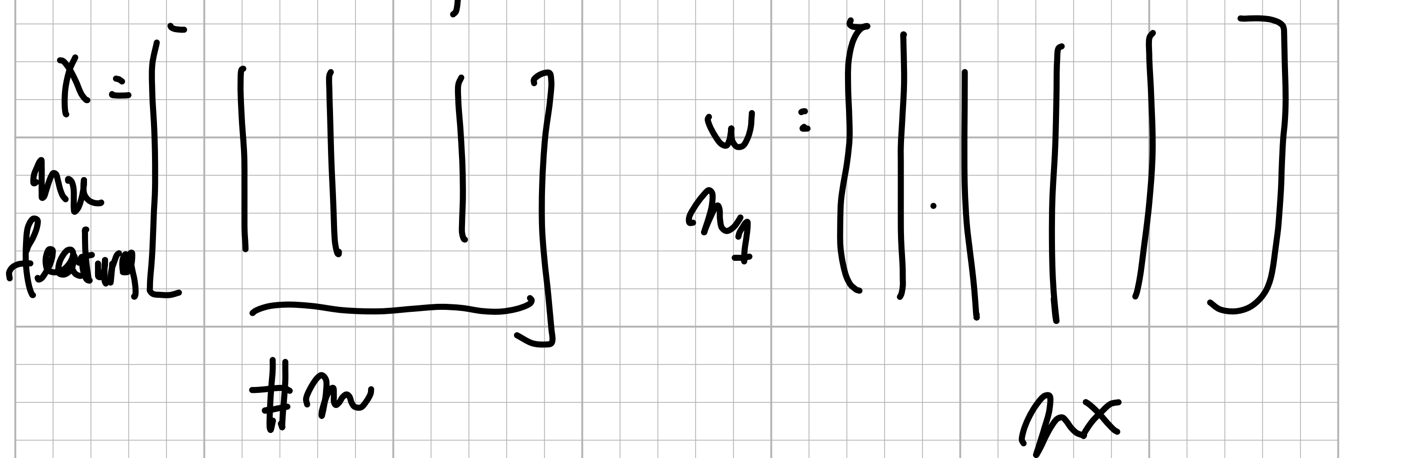
$\mathcal{L}(a, y) = -y \log a - (1-y) \log(1-a)$

$da = -y \cdot \frac{1}{a} + (1-y) \cdot \frac{1}{1-a}$

$dz = da \cdot \frac{da}{dz}$

$= -y \cdot (1-a) + (1-y) \cdot a$

$= a - y$



n_x = # features

n_1 = # neurons

Toán PTNK
HCMUS CNTN
Bz7 0

tramho

Deep Neural

$$z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]}$$

$$a^{[l]} = g^{[l]}(z^{[l]})$$

Oracle

Intern 5th

CanadaAI

VisionAI

WQ

Course 2 : Improving Deep Neural Networks:

Hyperparameter tuning, Regularization and Optimization

Bias / Variance

"Inverted dropout"

Data Augmentation