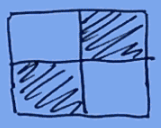
Neural Networks + CS506 Lecture (3/29/21)

* inherent limitations make it very far from working of brain activity

* Approach NOT from biological perspective
  ↳ Recall: LOGISTIC REGRESSION

• Given a 2×2 grid → find function f that can identify diagonal patterns (specifically, this one for now)



■ = $c_1 = 1$
□ = $c_2 = -1$ ↦

-1	1
1	-1

assume this is our early def. of diagonal

$$f\left(\begin{array}{|c|c|}\hline a_{00} & a_{01} \\ \hline a_{10} & a_{11} \\ \hline \end{array}\right) = \begin{cases} \text{yes, if } \begin{array}{|c|c|}\hline a_{00} & a_{01} \\ \hline a_{10} & a_{11} \\ \hline \end{array} = \text{  } \\ \text{no otherwise} \end{cases}$$

→ assign weights to each cell of the grid

$$\begin{array}{|c|c|}\hline w_1 & w_2 \\ \hline w_3 & w_4 \\ \hline \end{array} \quad \begin{array}{|c|c|}\hline a_{00} & a_{01} \\ \hline a_{10} & a_{11} \\ \hline \end{array}$$

$$w_1 a_{00} + w_2 a_{01} + w_3 a_{10} + w_4 a_{11} = \begin{cases} = b \text{ if diagonal found} \\ < b \text{ otherwise} \end{cases}$$

(if below some threshold, it's not a diagonal)

Ex] $\begin{array}{|c|c|}\hline w_1 & w_2 \\ \hline w_3 & w_4 \\ \hline \end{array} = \begin{array}{|c|c|}\hline -2 & 2 \\ \hline 2 & -2 \\ \hline \end{array}$ → apply these weights

$\times \begin{array}{|c|c|}\hline -1 & 1 \\ \hline 1 & -1 \\ \hline \end{array}$

$-2(-1) + 2(1) + 2(1) + -2(-1) = 8$ → this is b the solution for a diagonal

strange having this arbitrary #

* incorporate b into the eqn, so you can evaluate w/ reference to 0

$$w_1 a_{00} + w_2 a_{01} + w_3 a_{10} + w_4 a_{11} + b = \begin{cases} 0, \text{ diagonal} \\ < 0, \text{ NOT diagonal (otherwise)} \end{cases}$$

↳ translate vagueness of spec. + reflect that in the uncertainty!

* say, instead of c_1 and c_2, we want a continuum of colors → $[c_1, c_2]$
  ↳ what methods do we already have to apply to this case??

 translate grid into vector → $[a_{00}, a_{01}, a_{10}, a_{11}]^T$ → 0 or 1 } BINARY CLASSIFICATION

$[c_1, c_2]$ $[c_1, c_2]$ ...

$[0, 1]$
↳ prob. of being diagonal
$P(\text{diagonal})$

* LOGISTIC REGRESSION →

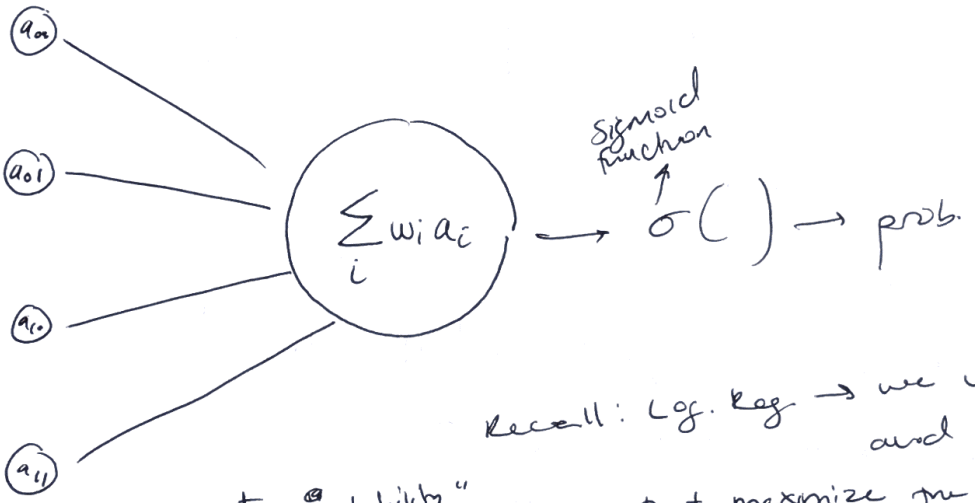
$$f\left(\boxplus\right) = \frac{1}{1+e^{-\vec{\omega}\vec{a}+b}}$$

$\vec{a} = [a_{00}, a_{01}, \cdots]$

b → BIAS

$\vec{\omega} = [\omega_1, \omega_2, \omega_3 \cdots]$

goal:



→ Recall from Log. Reg.

Recall: Log. Reg → we want the weights, $\vec{\omega}$, and bias, b.

that maximize the prob of having seen the data we saved

"product of the probability"

$$\max \prod_{i=1}^{n} P(y_i = 1 \mid x_i)$$

n = # data points

"smart trick" = multiply by y_i to transform into single eq

cost function (sum over all data)

$$= \min -\frac{1}{n} \sum_{i=1}^{n} \left[y_i \log\left(\frac{1}{1+e^{-wx_i+b}}\right) + (1-y_i) \log\left(1 - \frac{1}{1+e^{-wx_i+b}}\right) \right]$$

y_i = prob = 1 → $P(y_i = 1)$

prob. $y_i = 0$ → $P(y_i = 0)$

hide complexity under the "cost" (which we want to minimize)

$\hookrightarrow$ what is COST a function of?

$\text{Cost}(w, b)$

$$= \min \text{Cost}(w, b)$$

* want a numerical process for obtaining this minimum → start @ random w and random b

↳ make steps in the right direction.

$f\left(\boxplus\right) = \frac{1}{1+e^{-\vec{w}\cdot\vec{a}+b}}$ $[a_{00}, a_{01}, \cdots]$

b → BIAS, accounts for consistent/systemic dimming/brightening of the grid (something consistently less bright)

$[w_1, w_2, w_3 \cdots]$

goal: find weights that lead to right amount of activation of f that we either meet a threshold or not

---

→ suppose we start @ w_0 = random weight

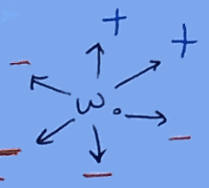
→ look @ Cost() around w_0 (all directions surrounding)

* + (pos) — (neg)

↳ pick best direction to minimize cost

↳ REPEAT ...

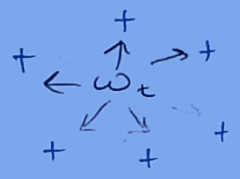
*some more negative than others



want cost to be as small as possible

⇒ clearly the best "nudge" to give w_0 & improve the cost

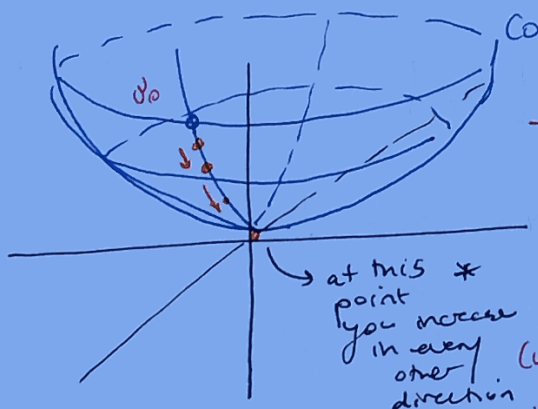
→ want to reach a point w_ϵ that looks like this



increases everywhere you go ... cannot further minimize *

(find some sort of min, maybe a local min)

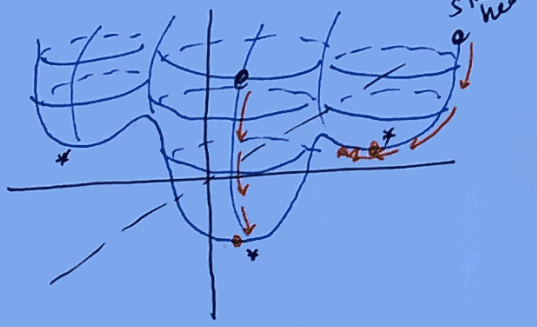
EXAMPLE



→ A basic function

↳ at this * point you increase in every other direction (w_ϵ) ↘ local minima

EXAMPLE — more complicated



↳ at best you find a local minimum. *

- and there may be many local minima

↳ the one you find depends where you start

Q How do you know direction of the best nudge?

↳ highest downward rate of change. (need the derivative)

⇒ discussion of GRADIENTS

↓

(derivative in each direction)

Gradients: Best "nudge" should be in direction of steepest rate of change (decreasing)

Note: rate of change = derivative

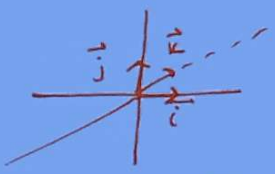
"gradient"

$$\nabla f(x,y,z) = \frac{df}{dx}\vec{i} + \frac{df}{dy}\vec{j} + \frac{df}{dz}\vec{k}$$

↳ global rate of change is the rate of change of → each orthonormal component (rate of change in each direction)

→ what is $\vec{i}, \vec{j}, \vec{k}$?

unit vectors for each direction in space



gradient = rate of change in direction of each coordinate

$(a\vec{i} + b\vec{j} + c\vec{k}) = [a, b, c]$

* unit length, orthonormal vectors

Example!

$$f(x,y) = \frac{3}{2}x^2 - y$$

* can already tell Δx affects f more than a Δy

$$\nabla f = \frac{df}{dx}\vec{i} + \frac{df}{dy}\vec{j}$$

$$= 3x\vec{i} + (-2)\vec{j} = 3x\vec{i} - 2\vec{j}$$

evaluate ∇f @ $p = (0,0)$

$$\nabla f_{(0,0)} = 3\cdot 0\,\vec{i} - 2\cdot\vec{j}$$

$$= -2\cdot\vec{j}$$

↳ gradient @ origin

"move two units away in j^th direction"

$$p' = \alpha \cdot \nabla f_{(0,0)} + p$$

$$= \alpha \cdot \begin{bmatrix} 0 \\ -2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -2\alpha \end{bmatrix}$$

$$f(p') = \frac{3}{2}(0) - 2\cdot(-2\alpha) = 4\alpha > f(p) = 0$$

↳ move α steps in direction of gradient, increase the value of function

gradient moves in most pos. direction of change

recall: we want opp., so add − to ∇f and we'll find Min

sanity check that we increased f

say $p'' = \alpha\begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

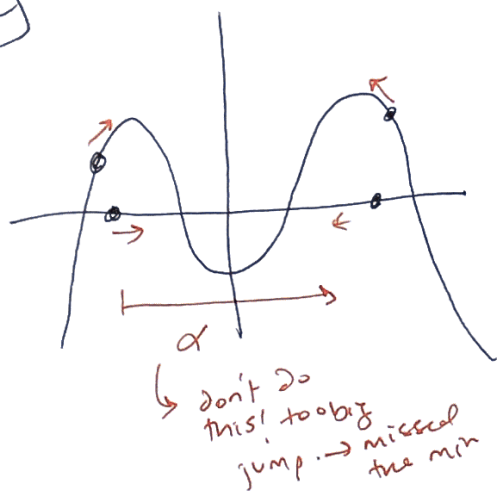
$$= \begin{bmatrix} \alpha \\ 0 \end{bmatrix} \qquad f(p'') = \frac{3}{2}\alpha^2$$

$$\begin{bmatrix} 4\alpha > \frac{3}{2}\alpha^2 \\ \text{or} \\ < \end{bmatrix} ?$$

↳ need to take SMALL steps so you can capture it all

↳ don't want to overshoot direction of min

Example



↳ don't do this! too big jump → missed the min

* α too big... end up bouncing back and forth (miss the min)

Gradient Descent ☆ → important

∇f → direction of steepest increase so $-\nabla f$ → direction of steepest decrease

1) Define α (tuning parameter) (not too small, not too big) → size of the step we want to take (how much of a nudge you want to give)

2) Initialize p to be random

3) $p_{new} = -\alpha \nabla f_{(p)} + p$ → update so $-\alpha$ steps in direction of gradient → update p

4) $p \leftarrow p_{new}$

5) Repeat 3 & 4 until $p \approx p_{new}$ (or gradient is zero)
(much easier constraint, numerically)

Projects: Early Insights Presentation — lightning talk! (~3 min @ most)

⇒ ☆motivation/background ☆progress ☆next steps → ☆limitations what challenges have we encountered?

↳ practice your pitch so it's FAST!!

⇒ powerpoint/visualizations
- slides — copy/paste from Derivatives

(more info this weekend)

" Why is your project awesome? "