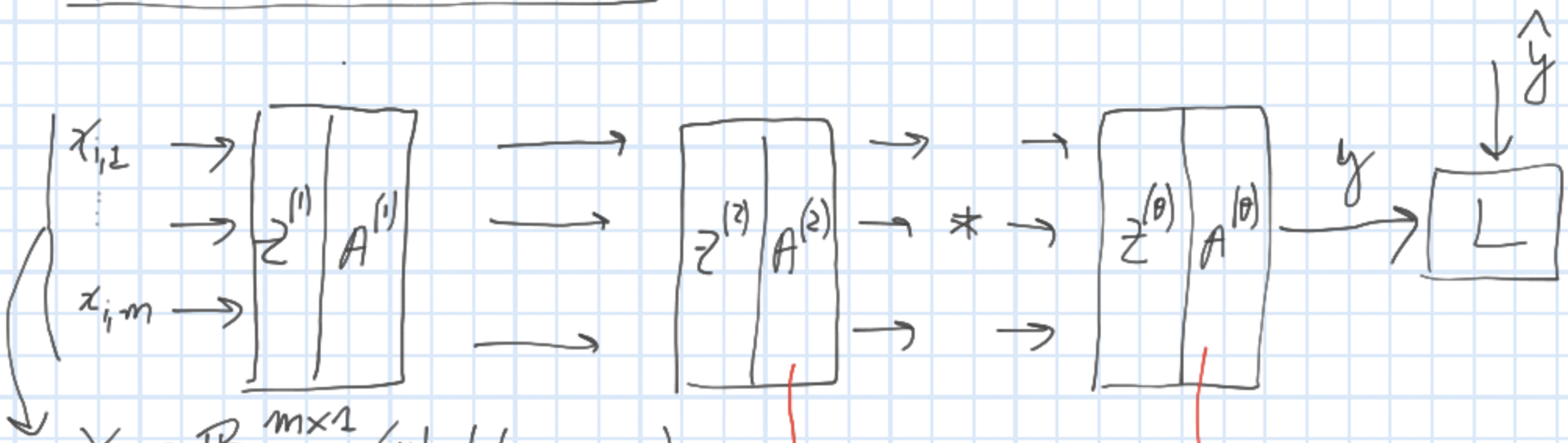
Clase 2:

* Funciones de activación



$X \in \mathbb{R}^{m \times 1}$ (#batch size → 1)

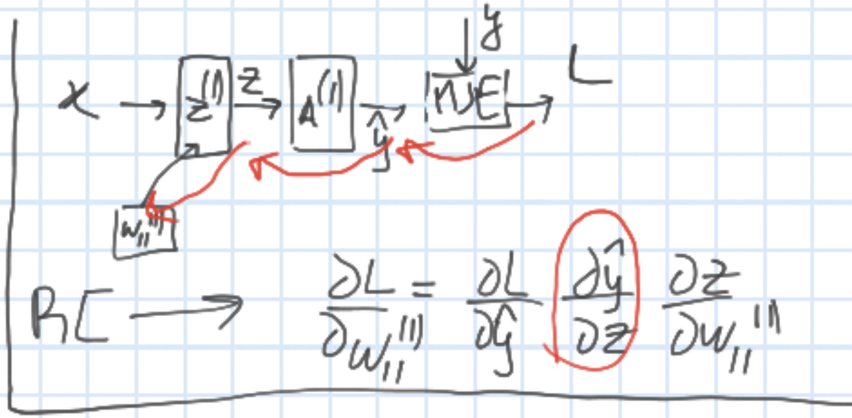
$X \in \mathbb{R}^{m \times b}$ (#batch size → b)

Sigmoid
ReLU
tanh

SOFTMAX → multi-clase
Sigmoid → clasificación binaria
Ninguna → regresión

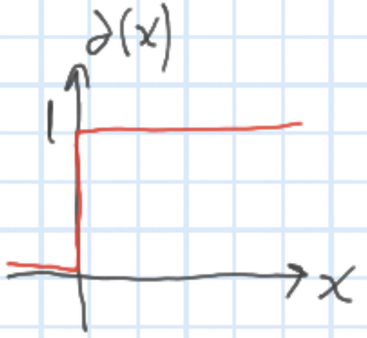
(1) No usar función de activación

$$x \rightarrow \boxed{z^{(1)}} \rightarrow \boxed{z^{(2)}} \rightarrow \hat{y} = f(x) = z^{(2)}\left(z^{(1)}(x)\right)$$



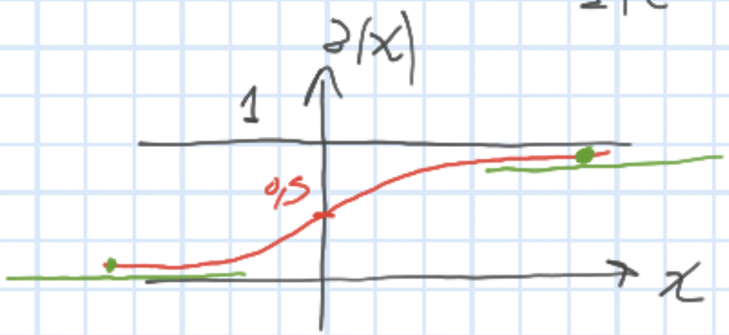
RC → $\frac{\partial L}{\partial w_{11}^{(1)}} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z} \frac{\partial z}{\partial w_{11}^{(1)}}$

(2)



$$\partial(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases} \longrightarrow \partial'(x) = \begin{cases} 0 & x > 0 \\ 0 & x < 0 \\ ? & x = 0 \end{cases} \longrightarrow w_i \leftarrow w_i - \alpha \frac{\partial L}{\partial w_i}$$

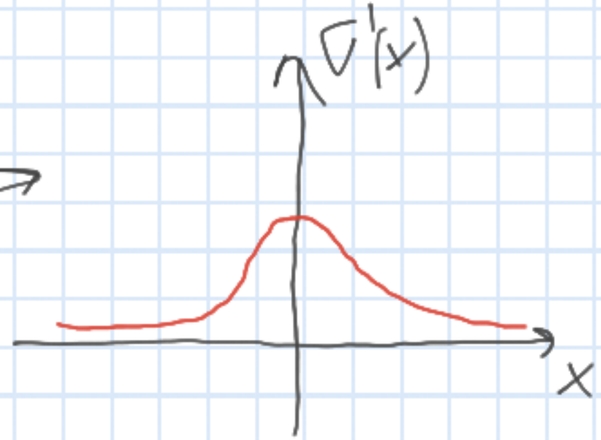
(3) $\partial(x) = \sigma(x) = \frac{1}{1+e^{-x}}$



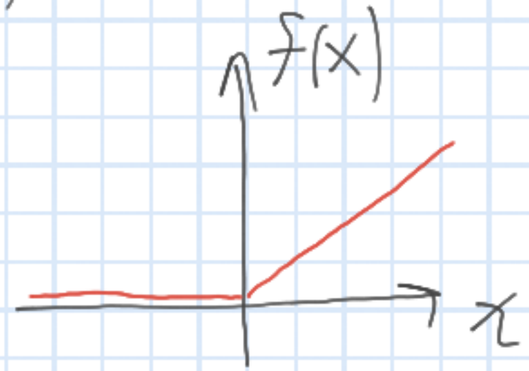
$$\partial'(x) = \frac{\partial \sigma(x)}{\partial(x)} = \sigma(x)(1-\sigma(x))$$

↳ Optimización lenta

Vanishing gradient



(4) ReLU

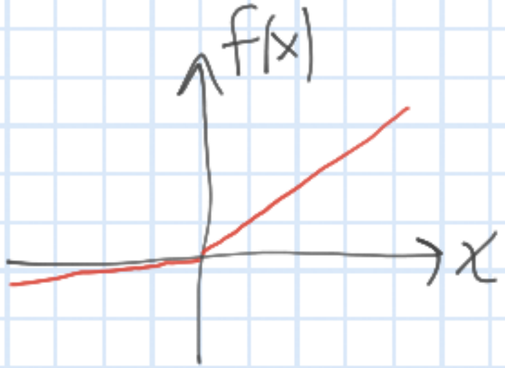


$$f(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases} \rightarrow f'(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$

$$= \max(0, x)$$

↳ Para $x < 0$, la derivada es zero

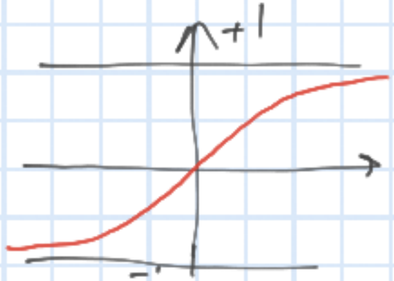
(5) Leaky ReLU



$$f(x) = \begin{cases} x & x > 0 \\ \partial x & x \leq 0 \end{cases} \rightarrow f'(x) = \begin{cases} 1 & x > 0 \\ \partial & x \leq 0 \end{cases}$$

↳ $\partial = 0{,}01$

(6) $\tanh(x) = 2\sigma(x) - 1$



* Funciones salida + Loss Function

$$\begin{array}{c|c} x_{1,1} \cdots x_{1,m} & y_1 \\ \vdots & \\ x_{n,1} \cdots x_{n,m} & y_n \end{array} \rightarrow f_{Y_i|\bar{X}_i}(y_i|\bar{x}_i)$$

$\bar{x}_n$: y_n

$$L(\bar{x}_1, \dots, \bar{x}_n, y_1, \dots, y_n) = \mathbb{P}\left(Y_1 = y_1, Y_2 = y_2, \dots, Y_n = y_n \,\middle|\, \bar{X}_1 = \bar{x}_1, \dots, \bar{X}_n = \bar{x}_n\right)$$

$$= \prod_{i=1}^{n} \mathbb{P}(Y_i = y_i | X_i = x_i) \rightsquigarrow \prod_{i=1}^{n} f_{Y_i|\bar{X}_i}(y_i|\bar{x}_i)$$

* Clasificación Binaria: $\bar{x}_i \rightarrow [z^{(1)}] ** [z^{(n)}] \rightarrow [\sigma] \rightarrow \hat{y}_i \rightarrow [\text{BCE}]$ ($y_i \in \{0,1\}$)

* Regresión: $\bar{x}_i \rightarrow [z^{(1)}] ** [z^{(n)}] \rightarrow \hat{y}_i \rightarrow [\text{MSE}]$ ($y_i \in \mathbb{R}$)

↳ (1) Clasificación binaria → $Y_i|\bar{X}_i \sim \text{Bernoulli}(p_i)$, $p = \bar{w}^T \bar{x}_i$

⇓ Loss → BCE

↳ (2) Regresión → Loss → BCE

↳ $Y_i|\bar{X}_i \sim \text{Normal}(\mu_i, 1)$, $\mu_i = \bar{w}^T \bar{x}_i$

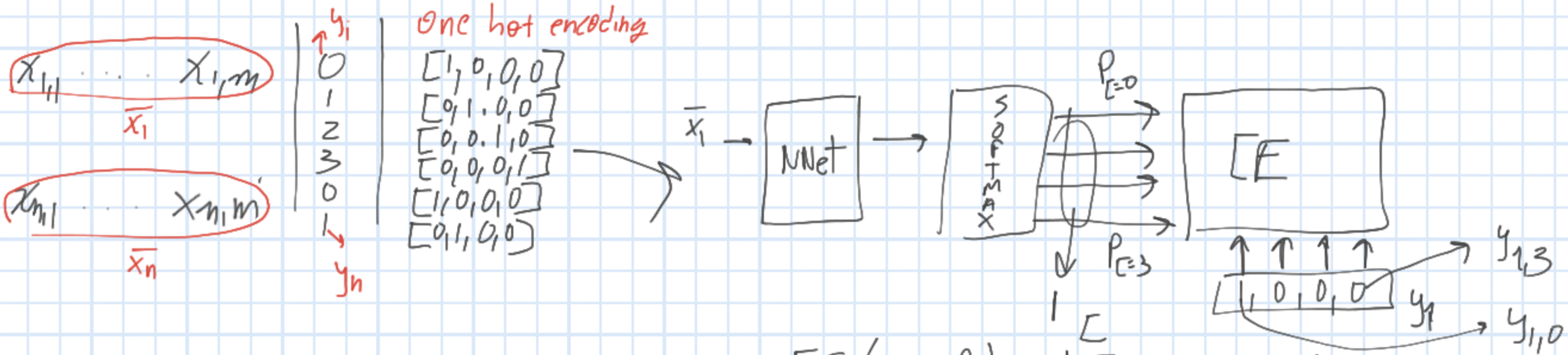
↳ Loss → MSE

* Clas. multiclase: C clases

$\bar{x}_i \rightarrow [z^{(1)}] ** [z^{(n)}] \rightarrow o_1, o_2, \dots, o_C \rightarrow [\text{SOFTMAX}] \rightarrow p_1, p_2, \dots, p_C \rightarrow [\text{CE}]$

$\bar{x}_1 \rightarrow y_1 = 2, C = 4 \rightarrow [0,0,1,0]$ one-hot-encoding $\rightarrow y_{1,1} \dots y_{1,C}$

$$p_i = \frac{e^{o_i}}{\sum_{j=1}^{C} e^{o_j}} \qquad \sum_{i=1}^{C} p_i = 1$$



$$CE(y_i, \hat{y}_i) = \sum_{j=0}^{C} y_{i,j} \log(P_{i,j})$$

$$= \sum_{j=0}^{C} y_{i,j} \log\left(e^{\theta_{i,j}} / \sum CE_i\right)$$

$$= \sum y_{i,j} \left[\log(e^{\theta_{i,j}}) - \log(\sum CE_i)\right]$$

$$= \sum_{j=0}^{C} y_{i,j} \left[\theta_{i,j} - \log \sum CE_i\right]$$

→ x → M1 → $\hat{y}$

$\hat{y}_{|x} = w_1 x^2 + w_2 x + w_3$ → params 3!

(roja)

Ejercicio Clase 2 → Usando SGD, encontrar w_1, w_2, w_3

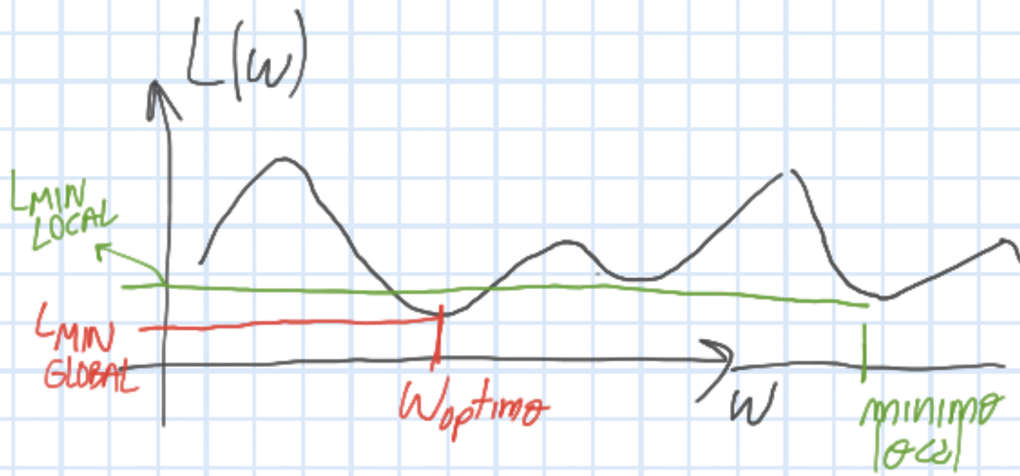
→ x → NNET → y → NNET → FIT CORRECTO

roja

→ x → NNET2 → y → OVERFITTING

→ Clase3 → PyTorch

Optimización



- Optimizar con gradiente $\nabla(L)$
  - Mini-BATCH (GD MB)
  - GD usando decaimiento
  - GD momento primer orden
  - AdaGrad, RMSProp
  - Adam (2014)
- Optimizadores que usan derivadas de orden superior $Hl(L)$
  - Newton
  - BFGS
  - L-BFGS
- Optimización mediante búsqueda
  - Genetico
  - Hill Climb

*
*

* Mini-Batch

epoch = 10

for e in epochs:

64 muestras
bs = 8

for b in batches → NNet

* $\bar{X}_b$
* $\hat{y}_b = \text{Forward}(\bar{X}_b)$
* Error $= L(y_b, \hat{y}_b)$

} * FORWARD

* $\nabla_{\bar{w}} = \frac{1}{\#bs} \sum_{i=1}^{\#bs} \ell(y_i, \hat{y}_i)$ * LOSS FUNCTION

bs = 1 → SGD
1 < bs < n → Mini-Batch
bs = n → GD

→ Backward

* $\bar{w} \leftarrow \bar{w} - \alpha \bar{G}$

1* ¿Cuantas veces actualiza $\bar{w}$ en una epoch?

$64 / bs = 64/8 = 8$

2* ¿Cuántas veces actualiza $\bar{w}$ en total?

$8 * ne = 8 * 10 = 80$

→ ¿HPs? ↔ α, epochs, #bs, ¿optimizador?

* AdaGrad

for e in epochs:
    for b in batches:
        * FORWARD
        * $\overline{G} \leftarrow$ BACKWARD
        * $\overline{r} \leftarrow \overline{r} + \overline{G} \odot \overline{G}$
        * $\overline{\Delta} \leftarrow -\frac{\alpha}{\sqrt{r}} \odot \overline{G}$
        * $\overline{w} \leftarrow \overline{w} + \overline{\Delta}$

* Adam (2014)

  * for e in epochs

    * for b in batch

      * Forward
      * Backpropagation
      * $\overline{G}$
      * $\overline{V} \leftarrow \rho_1 \overline{V} + (1-\rho_1)\overline{G}$ → momento 1er orden
      * $\overline{r} \leftarrow \rho_2 \overline{r} + (1-\rho_2)\,\overline{G} \odot \overline{G}$ → momento 2do orden
      * $\overline{\Delta} \leftarrow -\alpha / \sqrt{r}\, V$
      * $\overline{w} \leftarrow \overline{w} + \overline{\Delta}$