La distribución categórica.

$$\vec{p} = [p_1, \ldots, p_c]$$

$$0 \leq p_i \leq 1 \quad , \quad \sum p_i = 1$$

$\vec{p}$ vector de prob.

$\{\Delta, \circ, \square\}$

$\vec{p} = (0.5, 0.3, 0.2)$

$x \sim Cat(\{\Delta, \circ, \square\} \mid \vec{p})$

$prob[x = \Delta] = 0.5$

$prob[x = \circ] = 0.3$

...

$$Cat(V \mid \vec{p})$$

↳ {"Nathalia", "está", "estudiando", "matemáticas"}

$$w \sim Cat(V \mid (0.25, 0.25, 0.25, 0.25))$$

w = "está"

w = "estudiando"

Obj: aprender $\vec{p}$ a partir de los datos.

$$\vec{p}_\theta(\vec{x})$$

$\vec{x}$ ← input.

θ ↓ parámetros

↳ reg. polinomial

output

$y_\theta(\vec{x}) \in \mathbb{R}$

↳ reg. logística

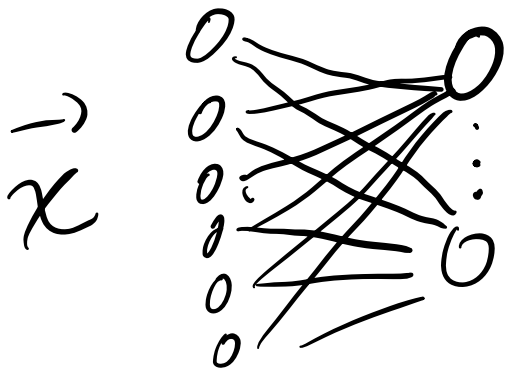
$\sigma_\theta(\vec{x}) \in [0,1]$

Prob. de pertenecer a una clase.

↳ $\vec{p}_\theta(\vec{x}) \in [0,1]^C$

núm. de clases

$$\vec{p}_\theta(\vec{x}) = \begin{bmatrix} p_1(\vec{x}) \\ \vdots \\ p_C(\vec{x}) \end{bmatrix}$$ un vector de probabilidad.



$$W_{c\times d}\vec{x}_{d\times 1} + \vec{b}_{c\times 1}$$

$$\sigma(x) = \frac{1}{1+\exp(-x)}$$ } Bernoulli, logística, sigmoidal.

$$\text{Softmax}\left(\begin{bmatrix} y_1 \\ \vdots \\ y_c \end{bmatrix}\right) = \left(\frac{\exp(y_c)}{\sum_{c'} \exp(y_{c'})}\right)_{c=1}^{c}$$

Softemax toma vectores cualquiera en $\mathbb{R}^c$, y devuelve vectores de probabilidad. torch.nn.Softemax()

$$\text{Softemax}\left(W\vec{x} + \vec{b}\right)$$

$W_{c\times d}$ $\vec{b}_{c\times 1}$

Dist	$f: \mathbb{R} \to P$
	enlace/link
Normal	id
Bernoulli (cat de 2 clases)	σ
Categórica	Softmax.

En torch:

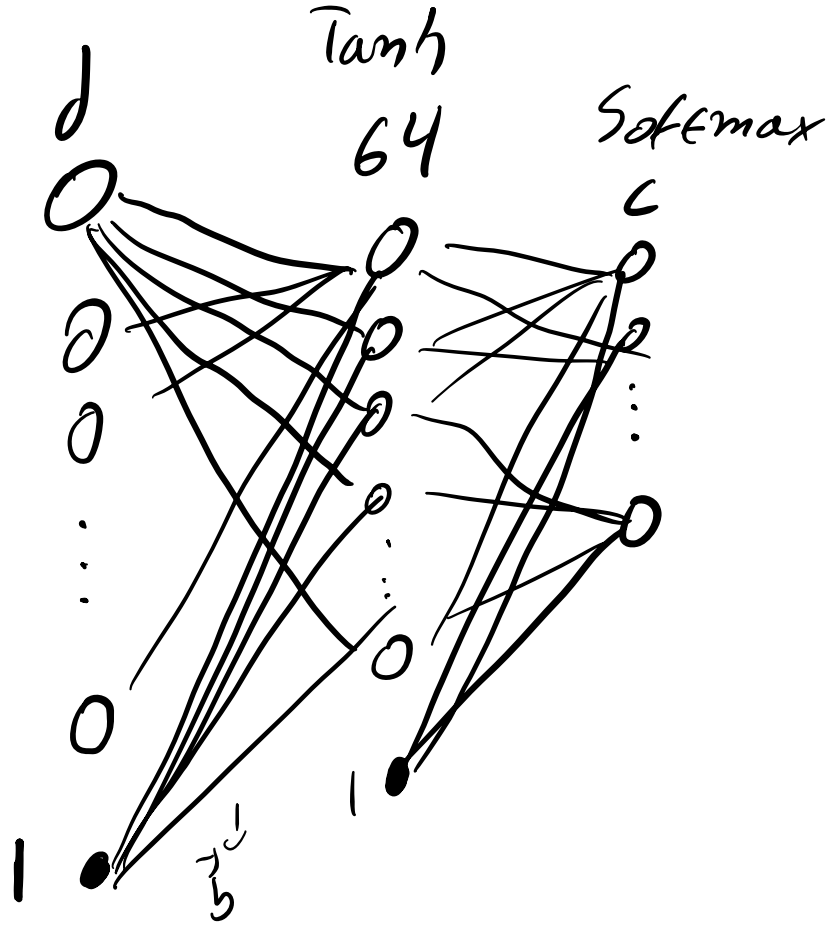
$\mathbb{R}^c \to \mathbb{R}^d$

x

```
Sequential (
    lineal (d, c),
    Softmax ()
)
```

} ej. lineal

```
Sequential (
    lineal(d, 64),
    Tanh(),
    lineal(64, c),
    Softmax()
)
```



Podemos crear un modelo

$$\vec{p}_\theta(\vec{x})$$

usando una red neuronal con activación softmax al final.

(1) Definir un modelo ✓

(2) función de error.

$(\vec{x}, 7)$

$\vec{x}$: imágenes

one-hot encoding

$$\vec{p}_\theta(\vec{x}_n) = \begin{bmatrix} p_0(\vec{x}_n) \\ \vdots \\ p_9(\vec{x}_n) \end{bmatrix} = \vec{p}_n \longleftrightarrow \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ 0 \end{bmatrix} = \vec{t}_n$$

10

Error: Entropía Cruzada multiclase.

$$-\sum_c t_c \log(p_c)$$

torch.nn.CrossEntropyLoss

$(\vec{p}, \vec{t})$

- Si estamos en lo correcto:

$t_7 = 1$, $p_7 \approx 1$

$p_{\neg 7} \approx 0$

$t_7 \log(p_7)^{\to 0} = 0$

$f(\vec{p}, \vec{t}) = 0$

$t_5 \log(p_5)^{\to 0} = 0$

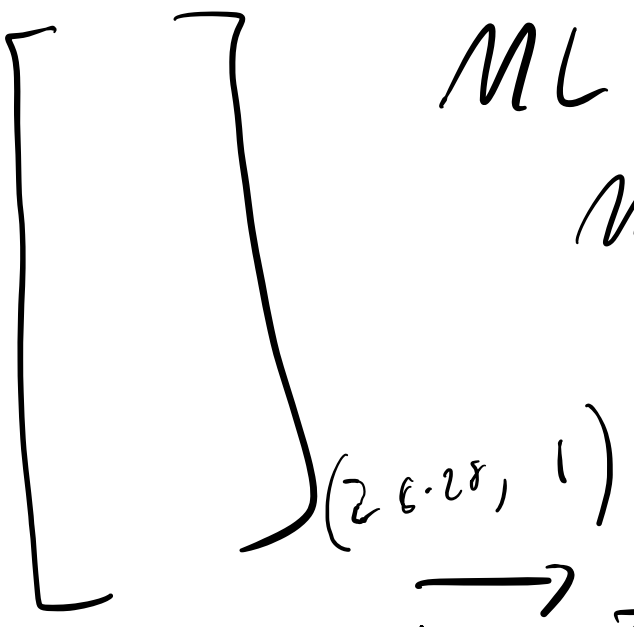
- Si estamos equivocados

$t_7 = 1$, $p_7 \not\approx 1$

$-t_7 \log(p_7)^{\to -\infty}$

MLP

MNIST.



0 1 2

(b, 28, 28)

- flatten(start_dim=1)

$(5, 28.28)$