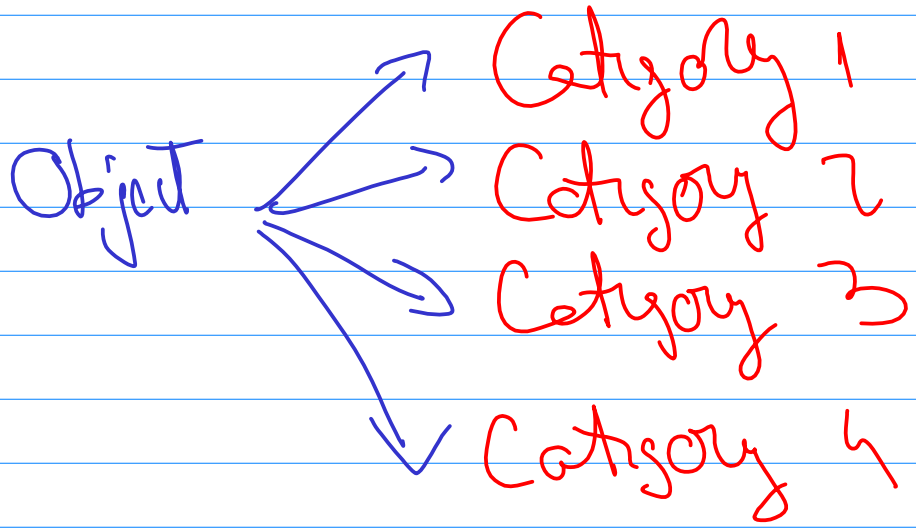
Course 3 - Classification

Object that can belong to one of a few fixed categories



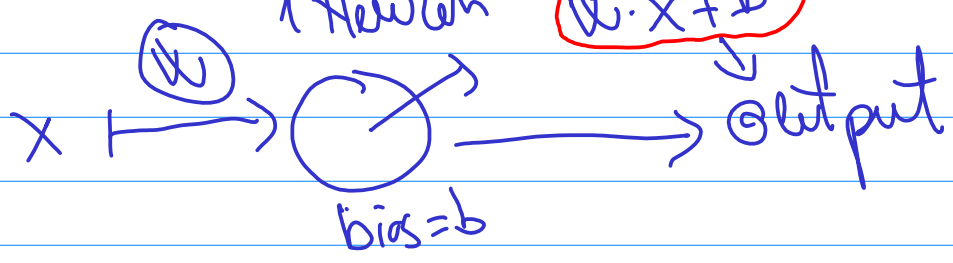
Binary ⇔ 2 categories.

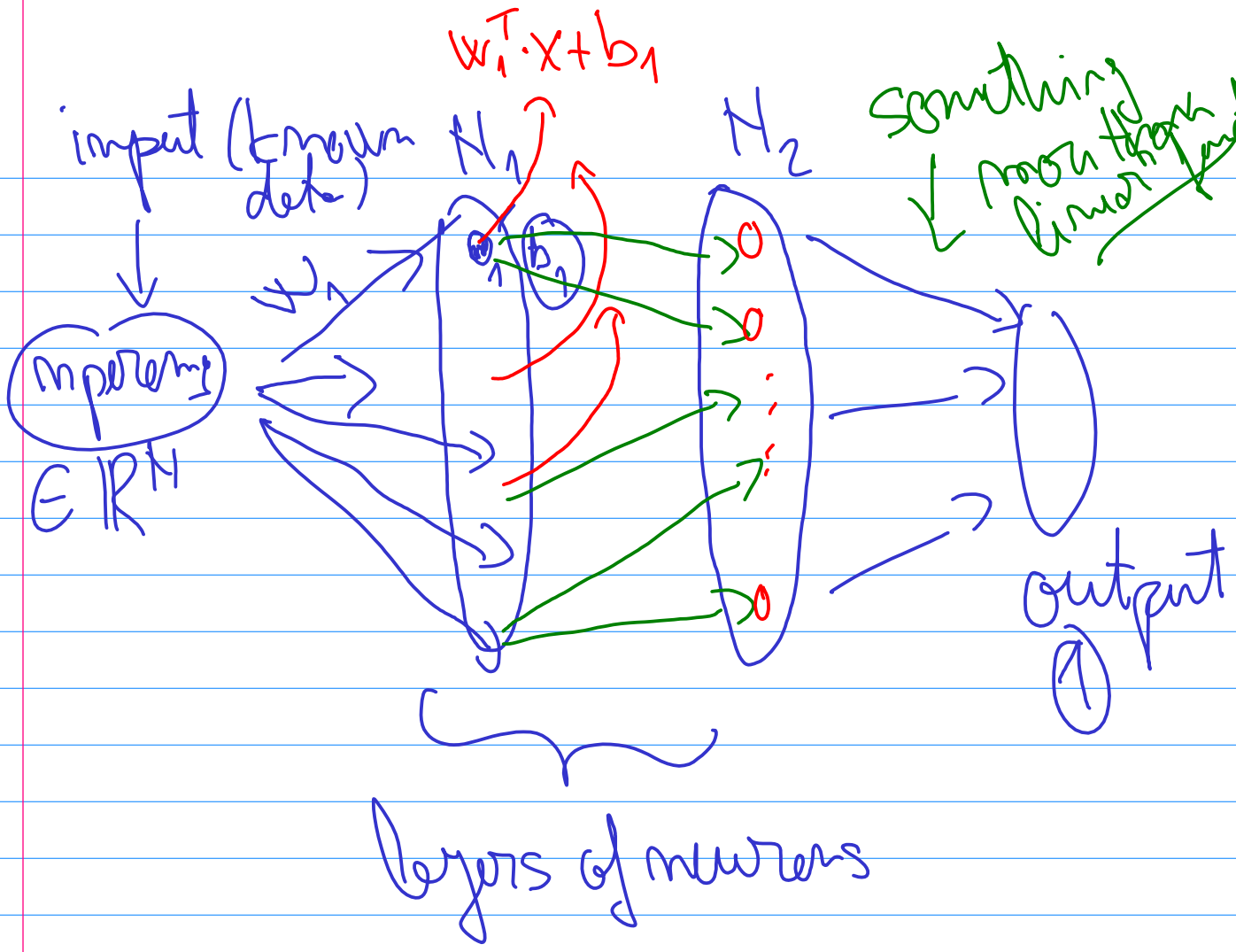
Multi-class ⇔ more than 2

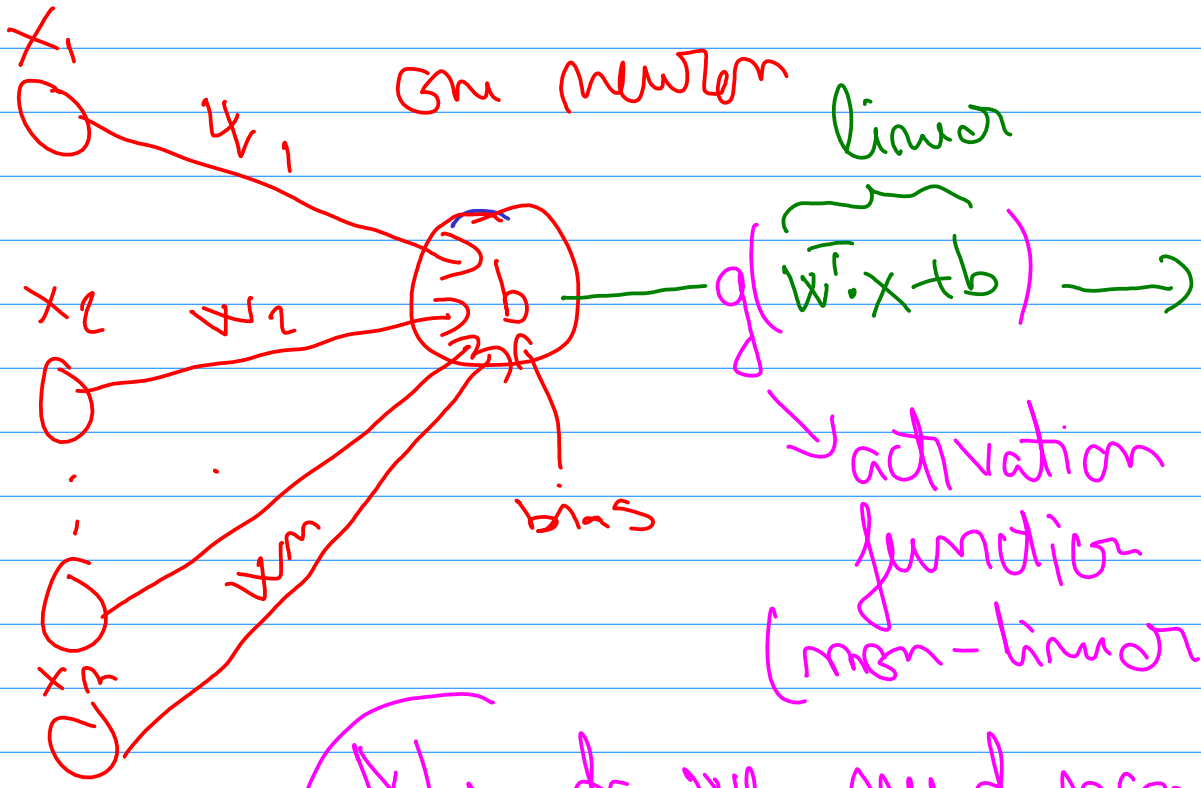
Loss function = a function which minimized aligns the model with the train data

What Loss func to use for classification?

Linear Regression (1D)

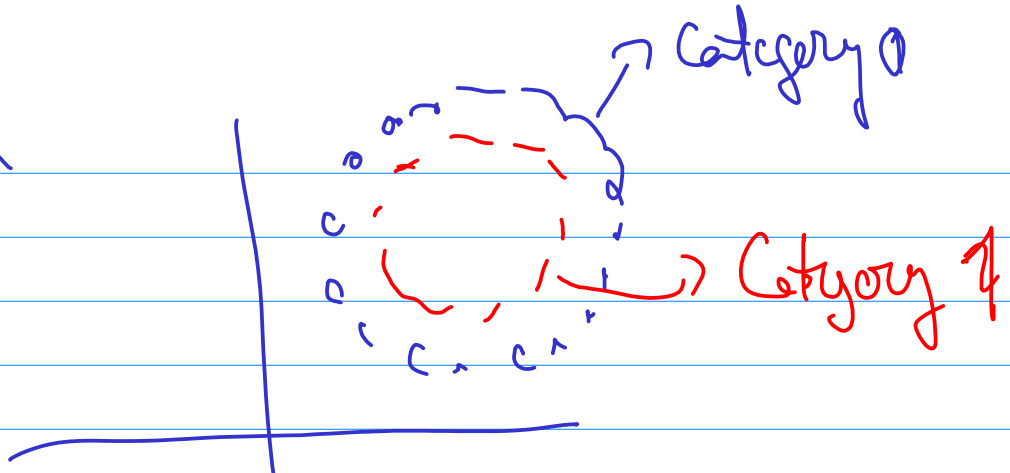






Why do we need non linearity?

Data

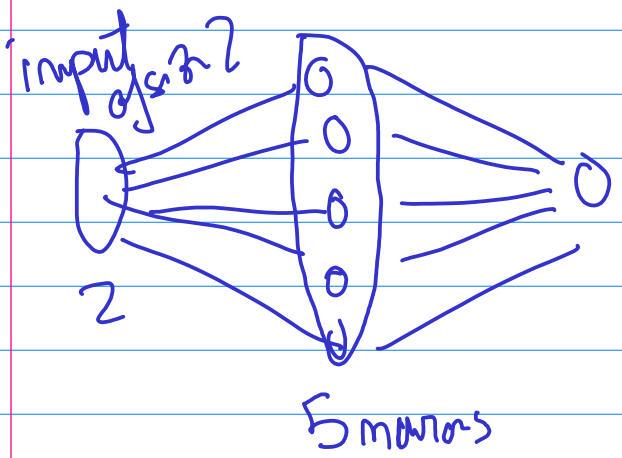


$$X = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \end{bmatrix} \qquad y = \begin{bmatrix} \text{label 1} \\ \text{label 2} \\ \vdots \end{bmatrix}$$

Circle Model V0

Dim

$2 \to 5 \to 1$



forward function

$\mathbb{R}^2$ $\mathbb{R}^5$

$X \to \text{layer1}(X)$ $\mathbb{R}$

$\to \text{layer2}(\text{layer1}(x))$

output

The dim of the output of layer L_i should correspond to the dim of input for L_{i+1}

Binary Cross entropy:

N number of obs. (samples)

y_i actual labels

p_i = probability given by the model for the i-th observation belonging to the class i

Model $(x) \xrightarrow{\text{gives}}$ a real number

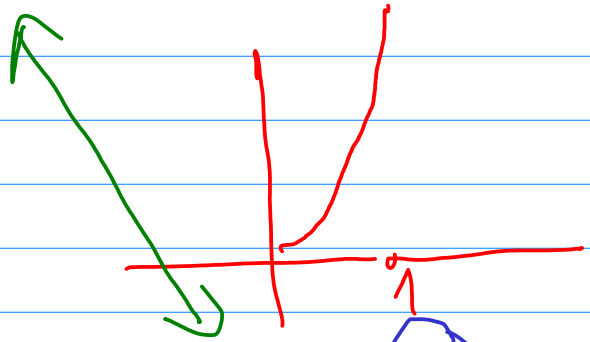
Map the reals onto $[0,1]$: sigmoid function

$$-\left[y_i \log p_i + (1-y_i)\log(1-p_i)\right] \rightarrow \text{!SMALL}$$

↓ training algorithm

$p_i \in (0,1)$

- if $y_i = 0 \Rightarrow BCE = -\log(1-p_i)$



BCE small $\Rightarrow p_i \rightarrow 0$

- if $y_i = 1 \Rightarrow BCE = -\log p_i$

BCE small $\Rightarrow p_i \rightarrow 1$

Accuracy test: y_{true} / $y_{predicted}$

Compute number of correct predictions

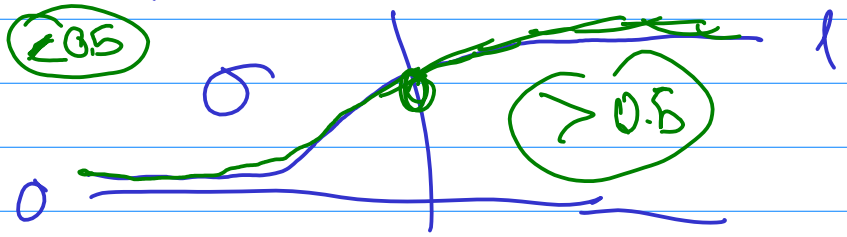
Sift of $y_{predicted}$.

50% accuracy → Bad → like a random model.

99% accuracy → Good

Logits = outputs (raw) of a neural network

Sigmoid function: turns nbs into [0, 1]

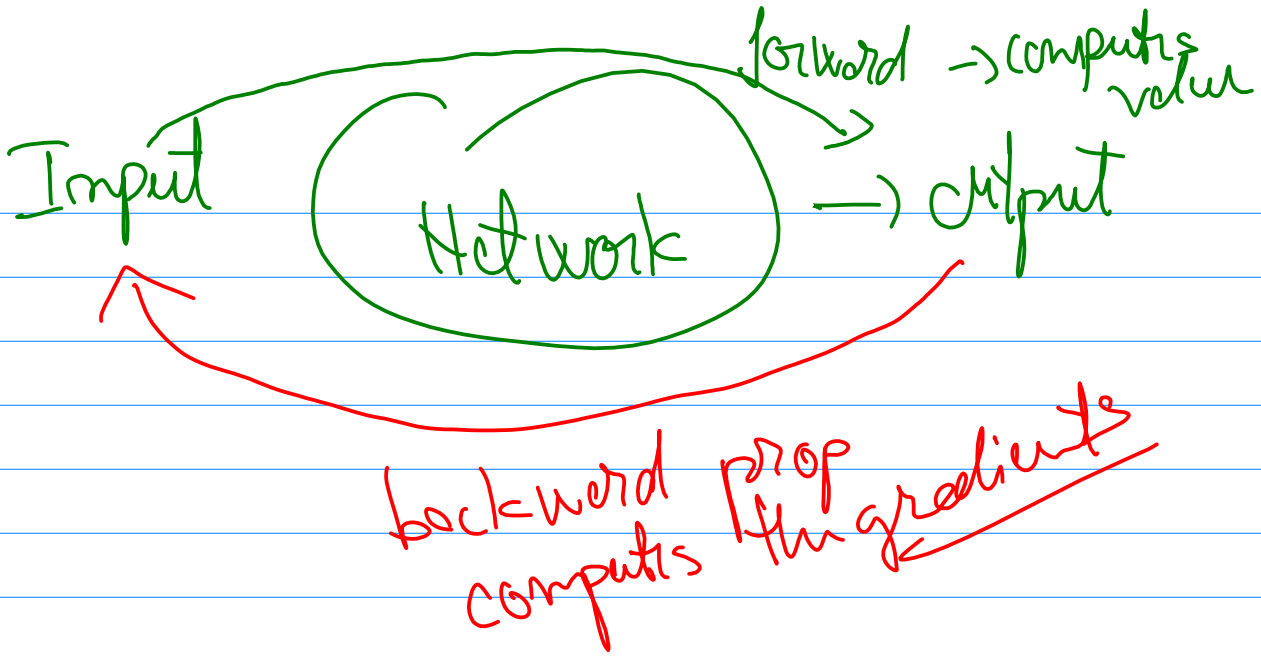


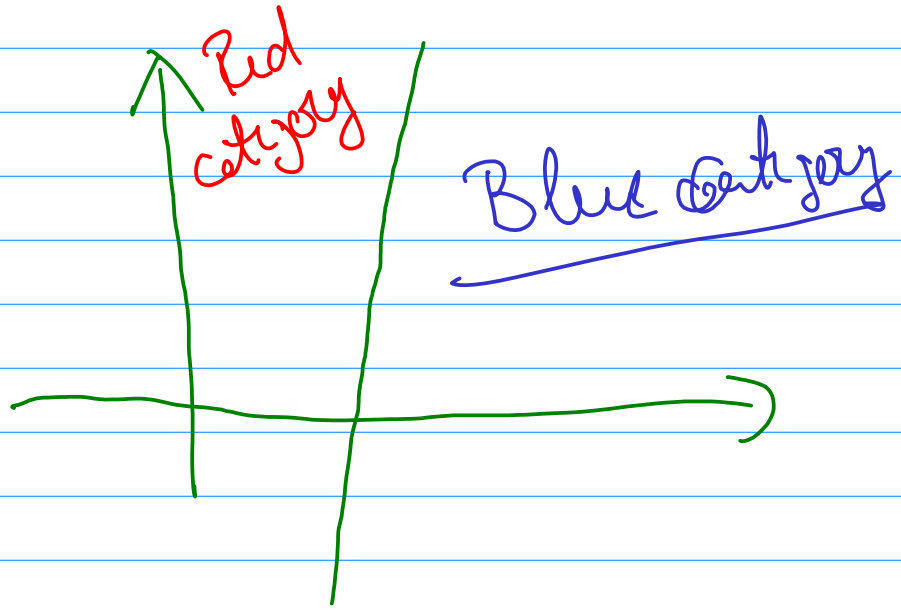
If $\sigma(\text{Model}(x)) > 0.5 \rightarrow$ Class 1

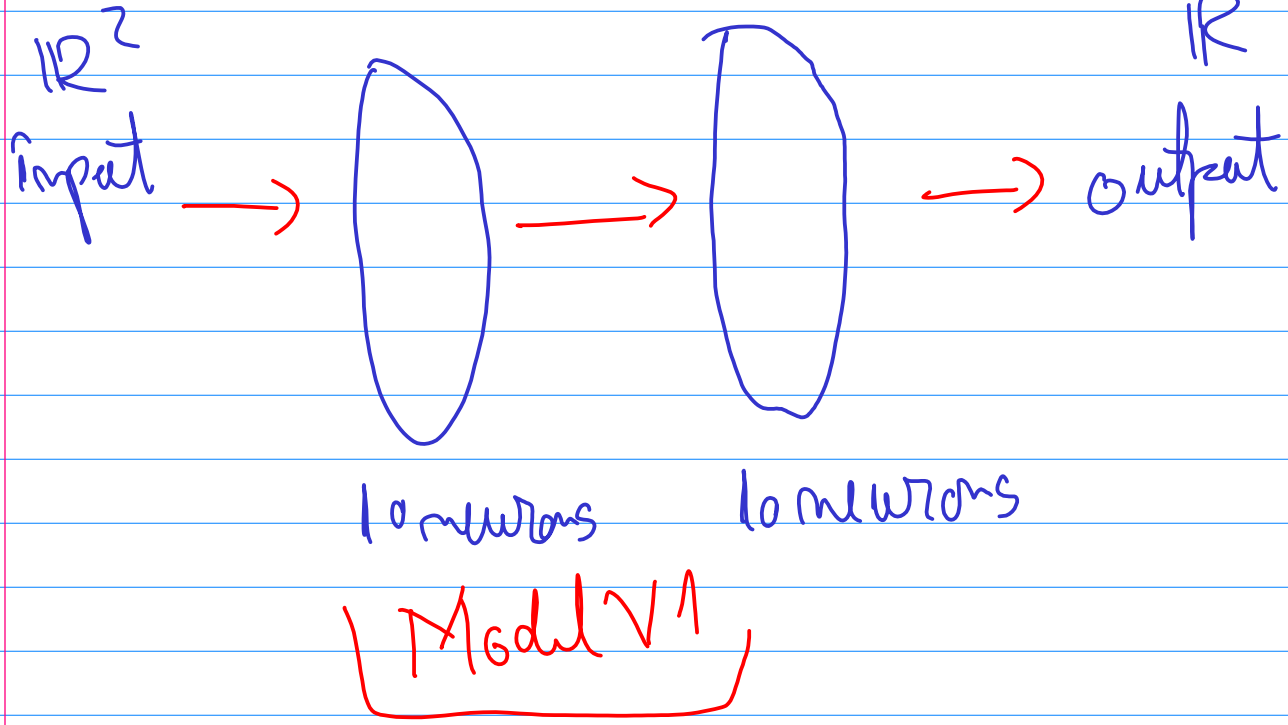
$< 0.5 \rightarrow$ Class 0

Output (IR) $\xrightarrow{\text{sigmoid}}$ [0, 1] $\xrightarrow{\text{round}}$ {0, 1}

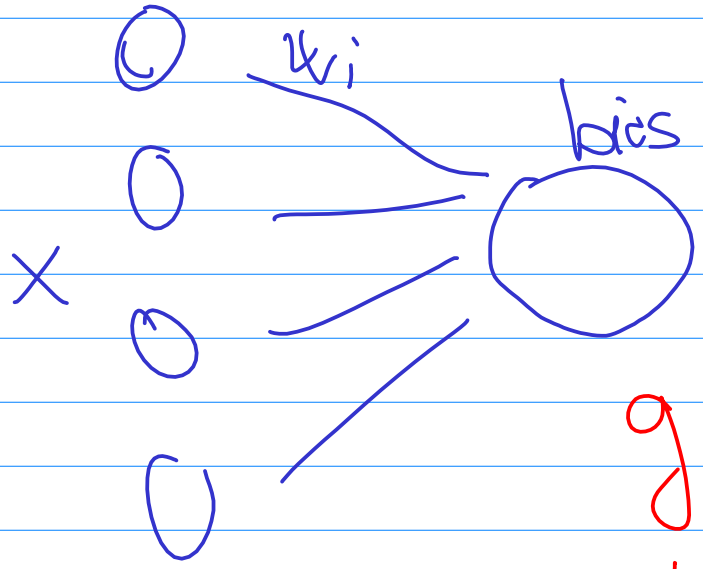
corresponds to what we have as labels.


forward -> computes value
Input
Network
-> output
backward prop computes the gradients


Red category
Blue category


$\mathbb{R}^2$
input
$\mathbb{R}$
output
10 neurons
10 neurons
Model V1

Without non linearities adding more layers does Nothing.



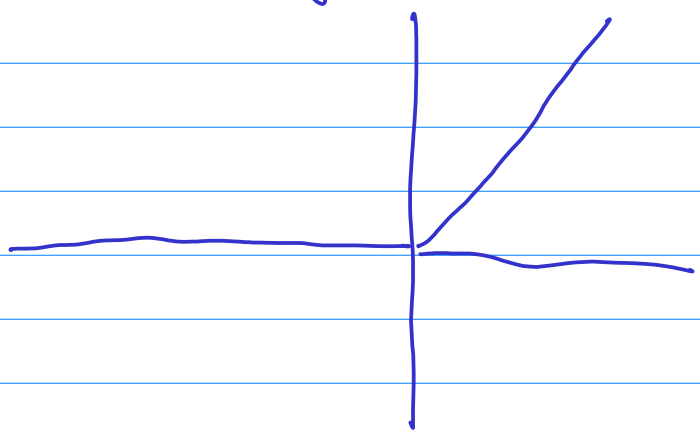
$$g(w \cdot x + b)$$

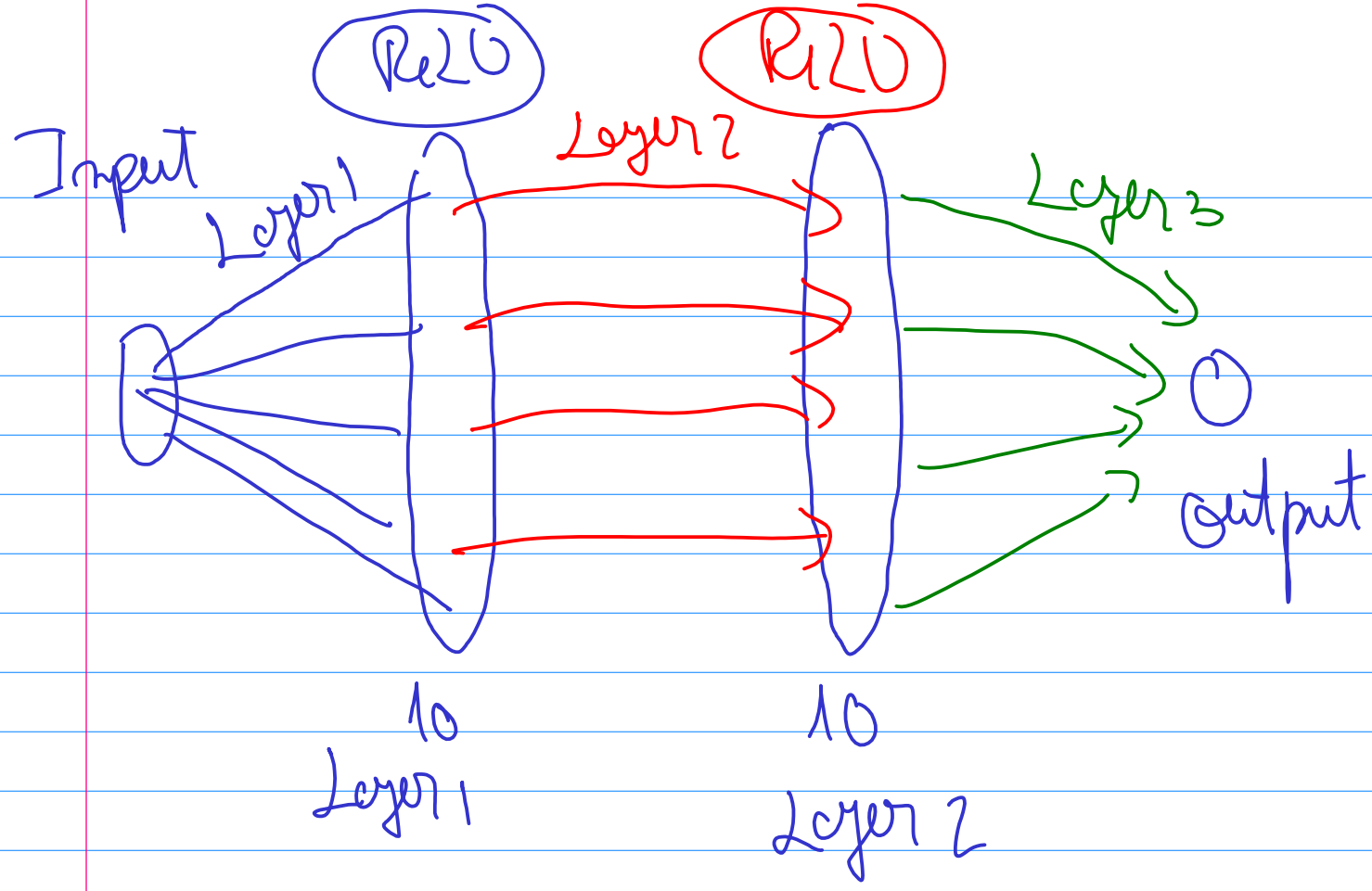
↓

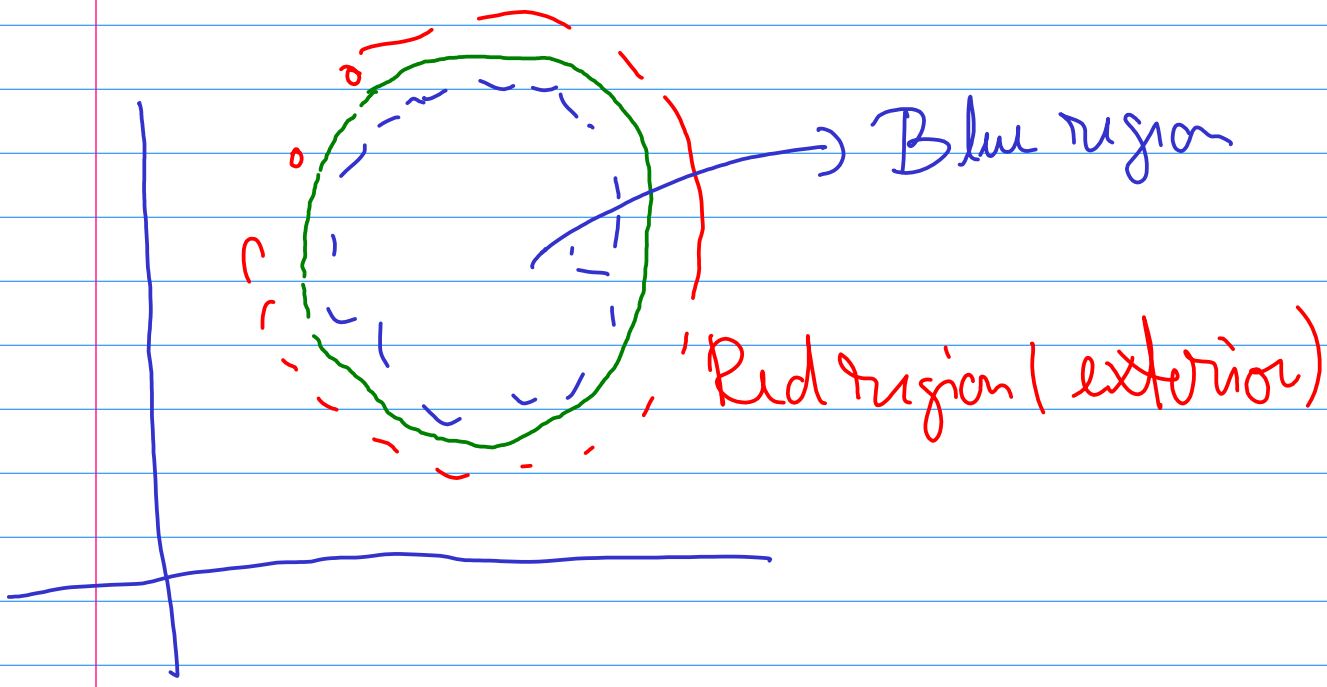
Nonlinearity applied

ReLU function:

$$\text{Relu}(x) = \begin{cases} 0 & : x < 0 \\ x & : x \geq 0 \end{cases}$$



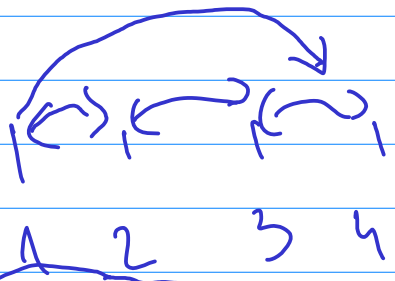

ReLU
ReLU
Input
Layer 1
Layer 2
Layer 3
0
output
10
Layer 1
10
Layer 2


Blue region
Red region (exterior)

4 Classes in $\mathbb{R}^2$

Proposed network

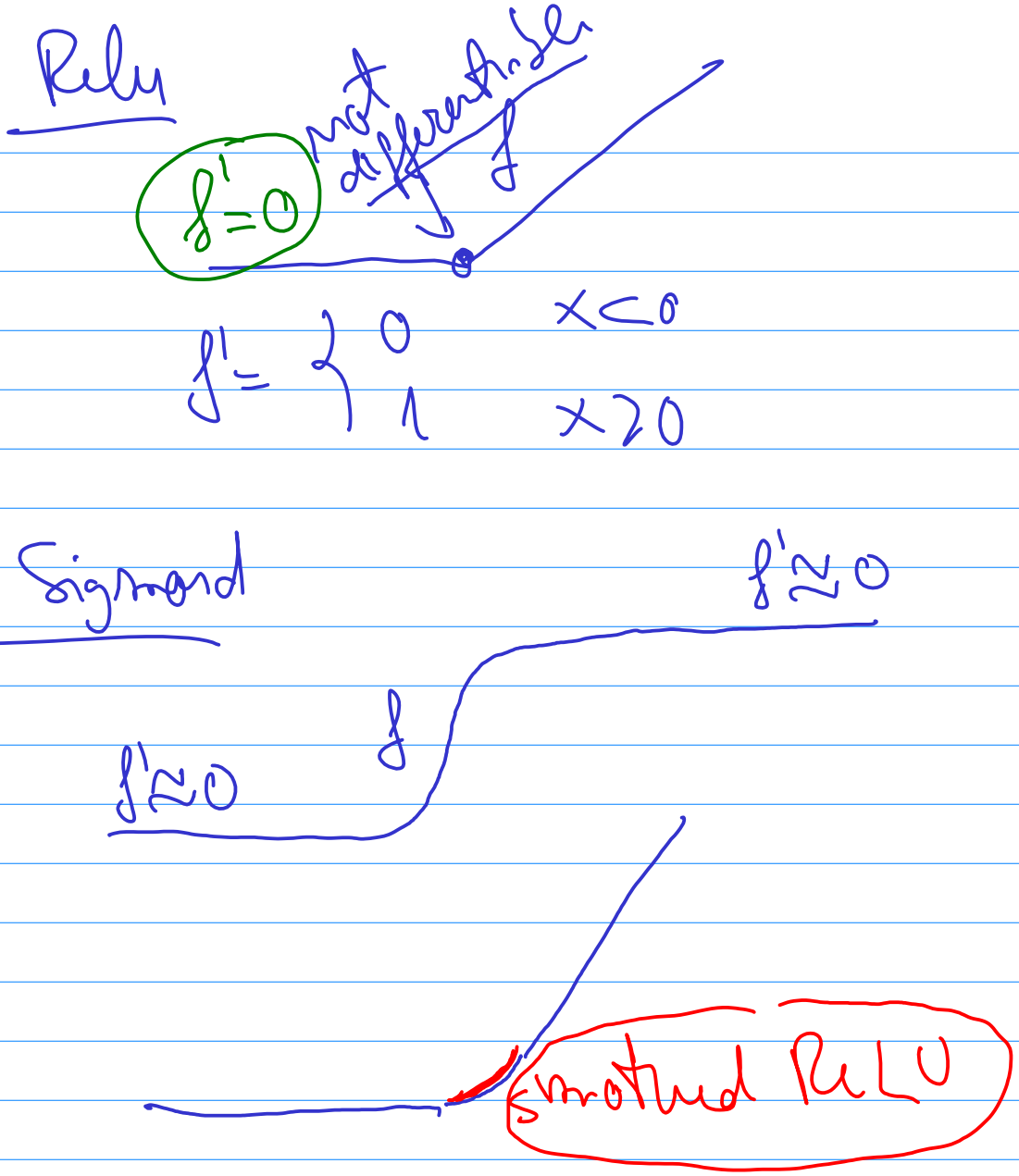
Why not consider Output in $\mathbb{R}$ (1, 2, 3, 4)



Not easy to jump from 1 to 4

Alternatively Output $\in \mathbb{R}^4$:

0,1	$P(X \in 1)$
0,5	$P(X \in 2)$
0,9	$P(X \in 3)$
0,2	$P(X \in 4)$


Relu
not differentiable
f'=0
f' = { 0 x<0
1 x>0
Sigmoid
f'≈0
f
f'≈0
smoothed ReLU