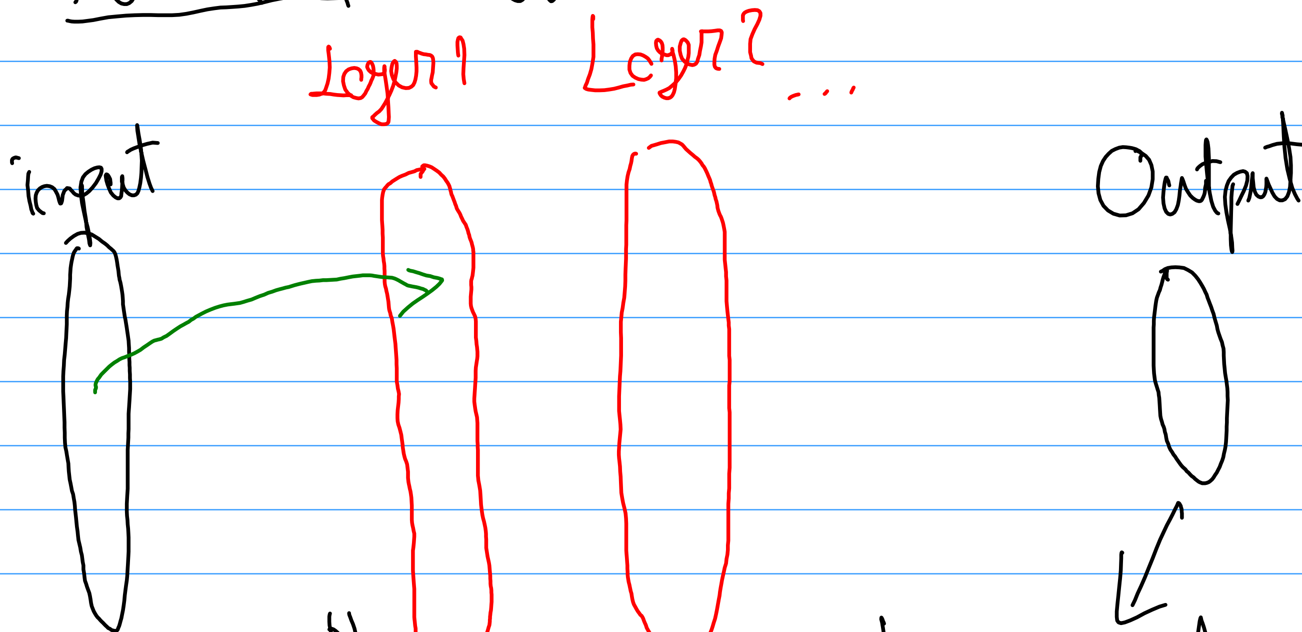
Course 4

Neural network

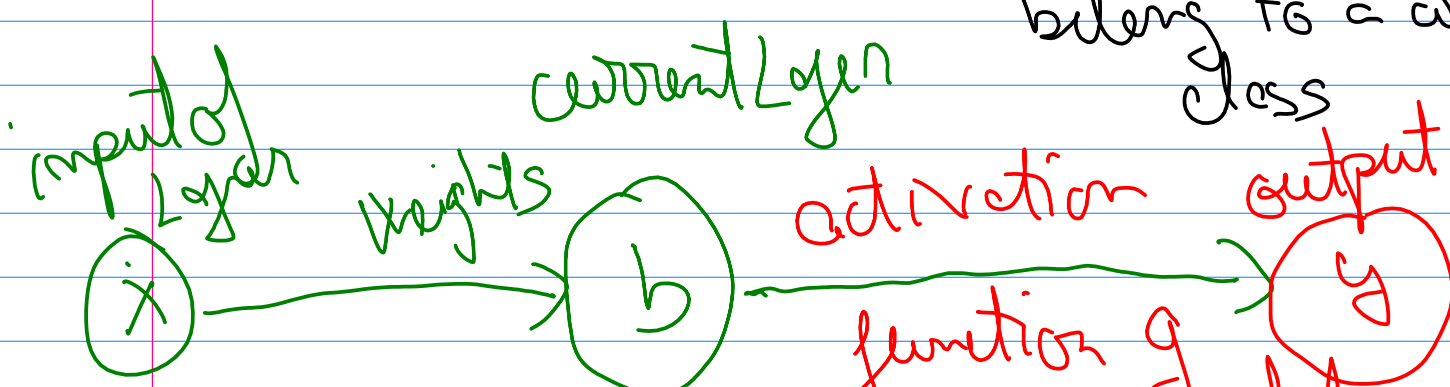


values in $\mathbb{R}^N$

any "tensor"
scalar, vector, matrix
3D matrix, etc.

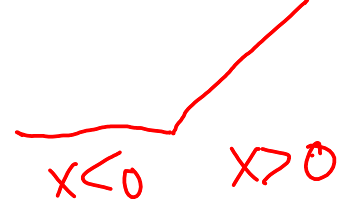
tensor: scalar
vector etc

Classification pb
probability to belong to a certain class

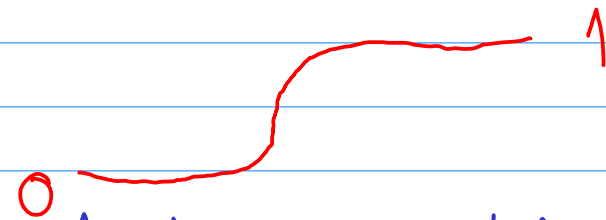


$$y = g(w^T \cdot x + b)$$

g → ReLU



g → Sigmoid



→ Deep Learning: find parameters of a given neural network such that it behaves "<u>like we want</u>"

→ <u>We need Data:</u>

<u>Structure</u>

$input_1$	$output_1$
⋮	
$input_N$	$output_N$

→ Split data into train set (80%) and test set (20%)

Testing: checking results of the model on things it hasn't seen before.

→ **Learning:**

→ Loss function:

Examples: $\sum_K \left| \text{model}(\text{input}_K) - \text{output}_K \right|$

(model: depends on w, b) (close)

→ The structure of the network is fixed before training.

→ Learning: optimization algorithm minimize the loss function

→ Non-linear activation functions are needed in order to capture "interesting" (non linear) phenomena

1. A simple example: learning a 1D function. Using a network on data which is very different than the training data will not provide useful results.

Exercise: learn a 1D function on [-1,1] and test it for values in a larger interval.

Neural networks → universal function approximators

$f(x) = x^2$

→ Data   Input x_i → Output x_i^2 | split into train/testing

→ define a model (Network)

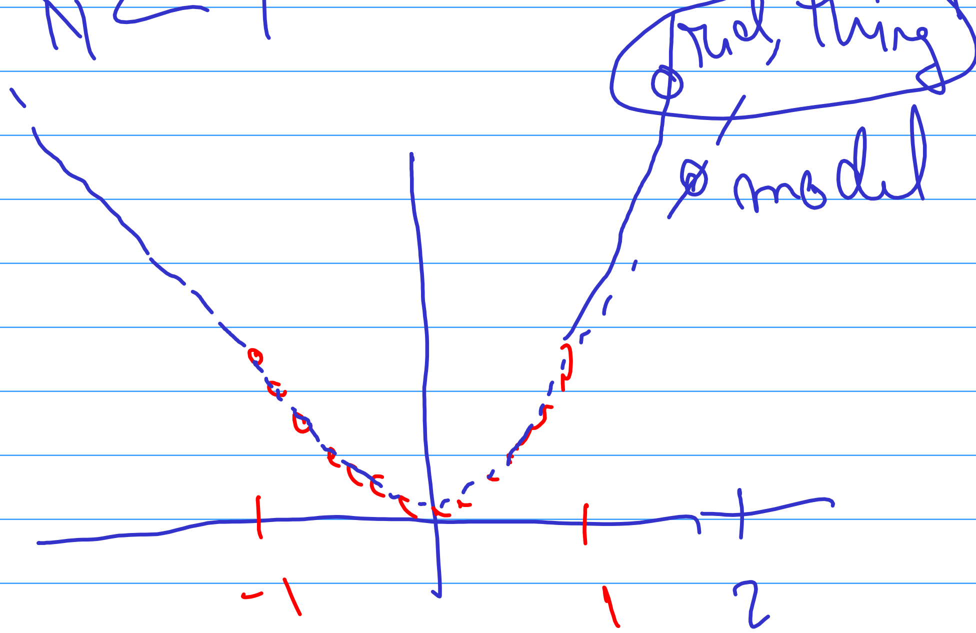
→ Training

→ Testing

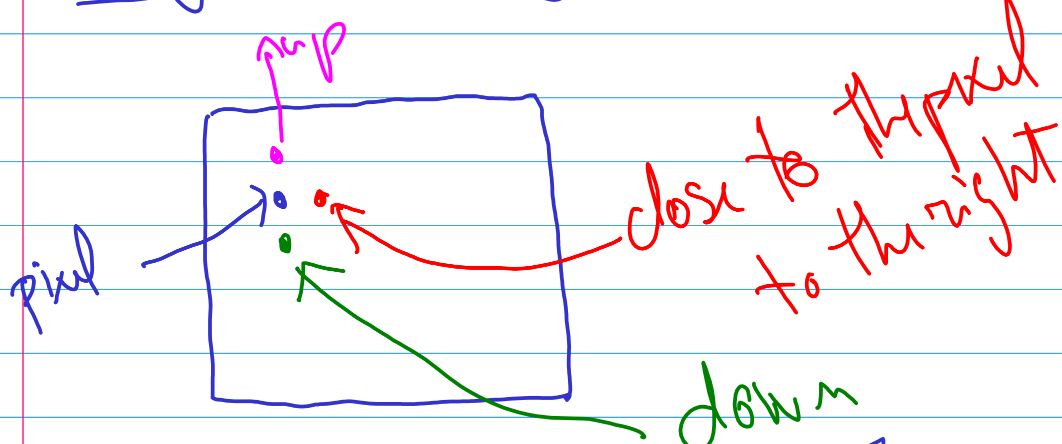
Dim input = 1
Dim output = 1

Mean Square error Loss

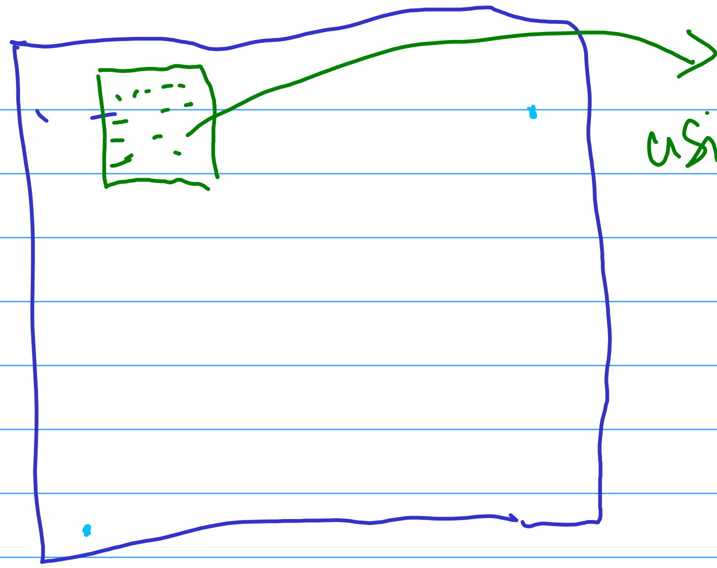
$$\frac{1}{N}\sum^{N} \left| model(x_i) - output_i \right|^2$$



Images are not just values



We need to interpret together pixels that are close geometrically.



aggregate pixels using a moving frame