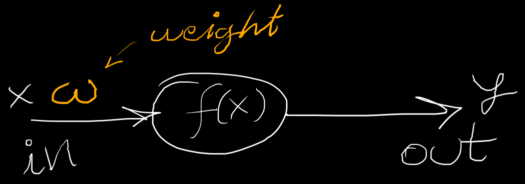
Ann Basic

* artifitial NeuralNetwork

①

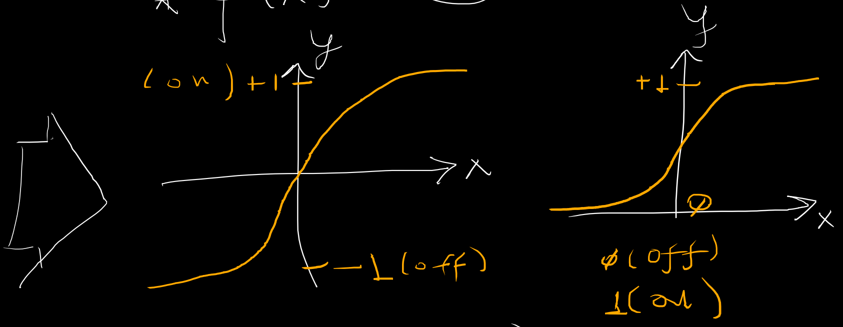
* An: artifihal Neuron
Perceptron



* Neuron's math model
- in = x (numbers) ← Known data
- out = y (numbers) ←
- ω = weight (number) ← unknown data!
- f(x): activation function

②

* f(x):

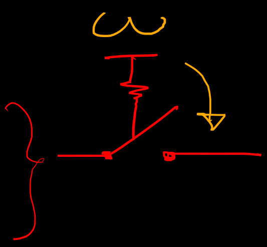


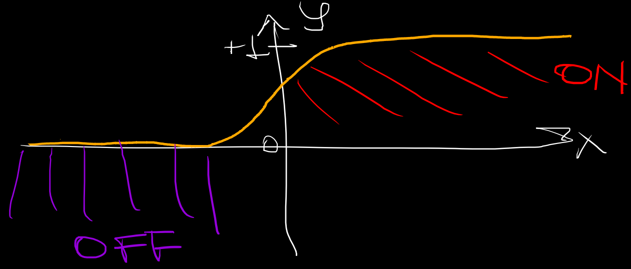
$$\underbrace{y}_{y(x)} = \underbrace{f(x \cdot \omega)}_{f(x \cdot \omega)}$$

③ f(x) Example

Sigmoid:

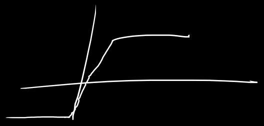
$$f(x) = \frac{1}{1 - e^{-x}}$$



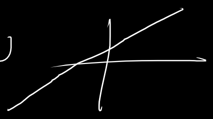


④ what is else?

* Sigmoide
* tanh



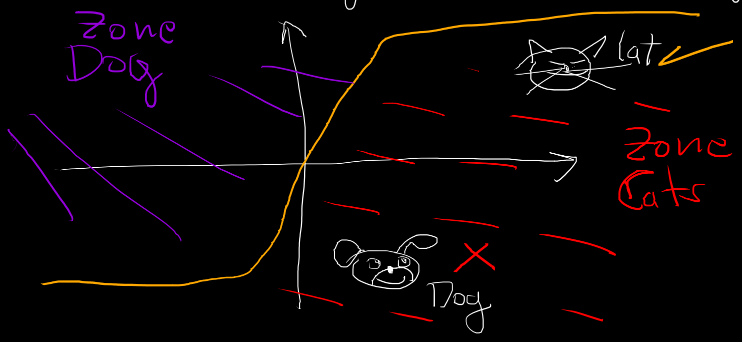
* ReLU

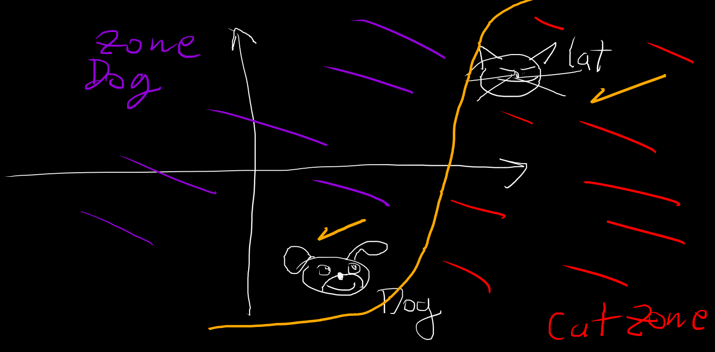


* maxout . . . .

etc !!

(5) Training an AN : Dog vs Cat Case





non-trained an
"w" is random

trained AN
"w" was computed

What is "train" ? = Solve w!!

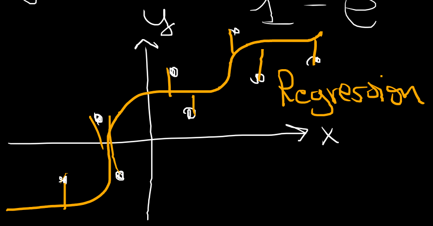
*(6) How to Solve ?

$$y = f(x \cdot w) = y = \frac{1}{1 - e^{-x \cdot w}}$$

$$y = \frac{1}{1 - e^{-x \cdot w}} \qquad y(x,w) = \frac{1}{1 - e^{-xw}}$$

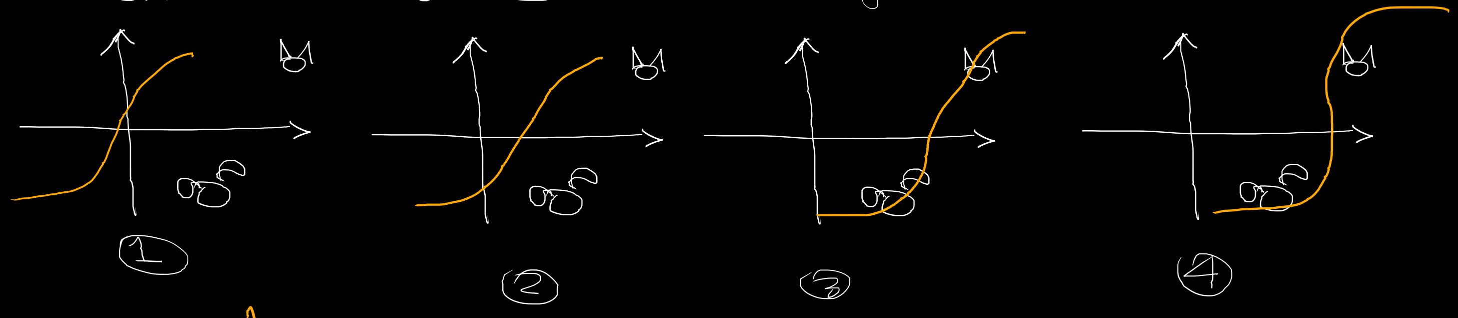
Input(x)	Output(y)
1	cat
2	Dog
3	Cat
4	Dog
5 .	.
6 .	.

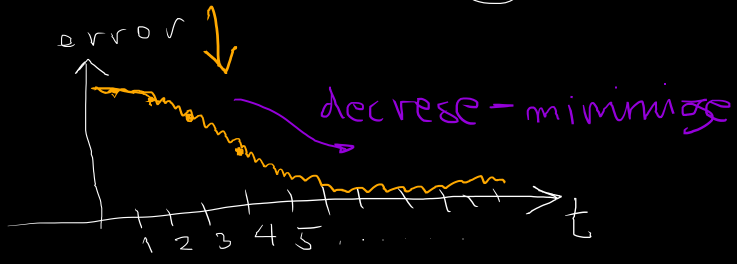
$$x = [\cdot \;\; \cdot \;\; \cdot \;\; \cdot \;\; \cdot]$$
$$y = [\cdot \;\; \cdot \;\; \cdot \;\; \cdot \;\; \cdot]$$



$$\|error\| = \|\underbrace{y(x,w)}_{real} - \underbrace{f(x \cdot w)}_{prediction}\|$$

* Error ⟶ Loss ⇨ Training is an iterative process





Training process ;

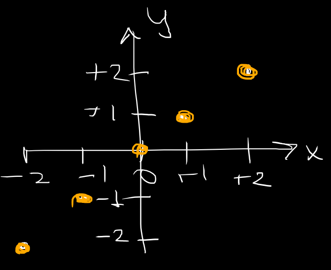
$$\|\underbrace{Loss(w)}_{min}\| = \|\underbrace{error(w)}_{min}\|$$

⑦ Example

$x \rightarrow \boxed{f(x)} \rightarrow y$

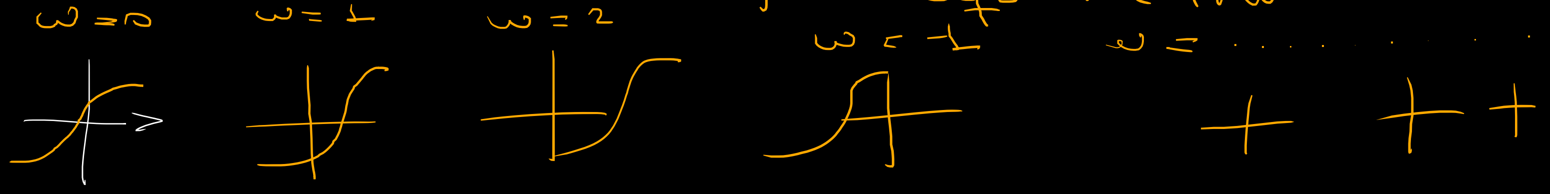
$$y = \frac{1}{1-e^{-x\omega}} = y(x,\omega)$$

x	y
−1	−1
−2	−2
0	0
+1	1
+2	+2



$$\left\{\begin{array}{lll} -1 = \dfrac{1}{1-e^{-1\underline{\omega}}} & -2 = \dfrac{1}{1-e^{-2\underline{\omega}}} & +2 = \dfrac{1}{1-e^{2\underline{\omega}}} \\ \emptyset = \dfrac{1}{1-e^{\emptyset\underline{\omega}}} & +1 = \dfrac{1}{1-e^{1\underline{\omega}}} & \end{array}\right.$$

$\nexists\ \omega$, all eqs are true



$y = f(x)$ / suppose $\omega = 1$

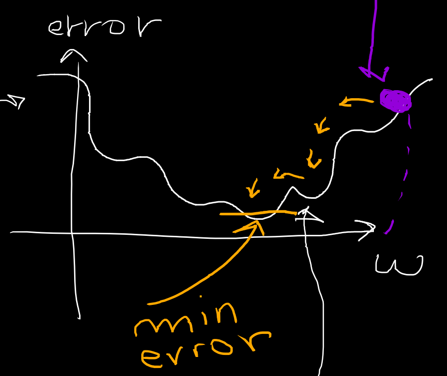
$$\downarrow$$

$$\underbrace{-1}_{\text{i want it}} = \frac{1}{1-e^{-1\omega}} = \frac{1}{1-e^{-1}} = \underbrace{-0{,}8}_{\text{predicted}}$$) when $\omega = 1$

error $= \|-1-0{,}8\| = 0{,}2$

For any "ω" $\text{error}(\omega) = y - \dfrac{1}{1-e^{x\omega}}$

$= \|y - f(x\cdot\omega)\|$



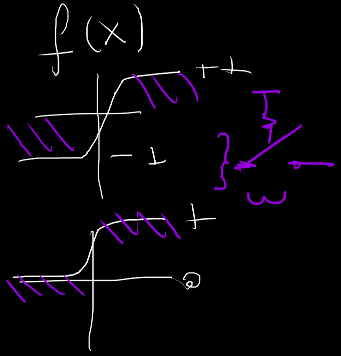
How to find min error ? Optimizer!!

* summary:

$\rightarrow \boxed{f(x)} \rightarrow$: perceptron

variables:

x
y } data known
f(x)
$\boxed{\omega}$ data known



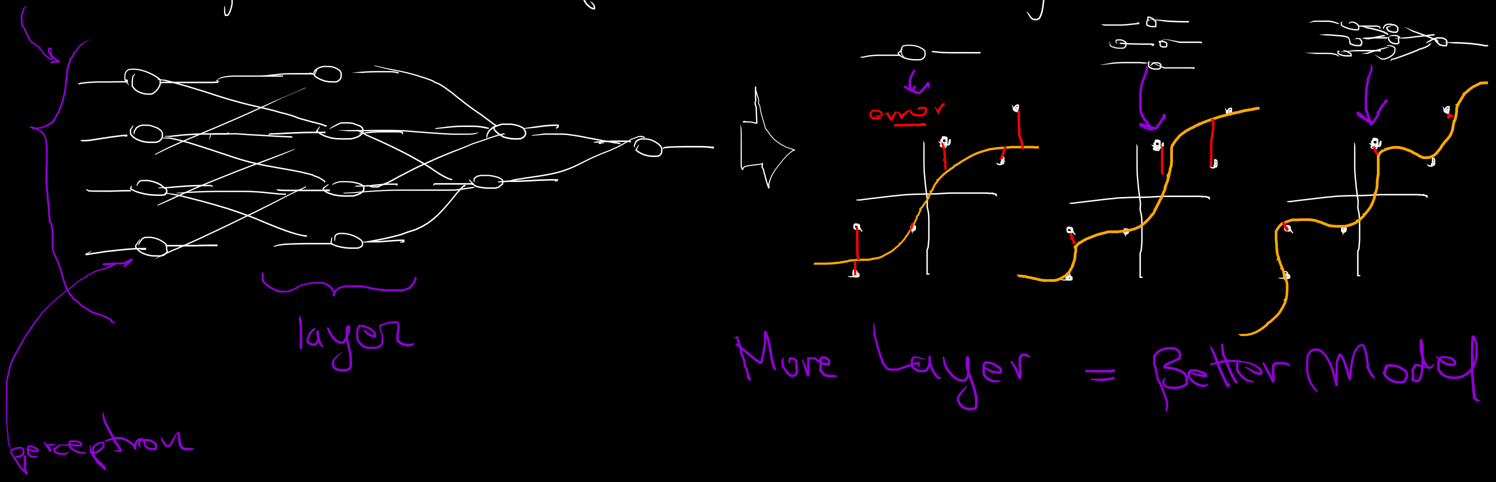
Train: loop iterative

Search ω optimo

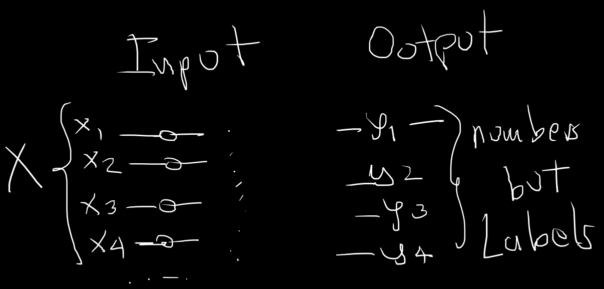
loss }
error } min ↓ decreases !!

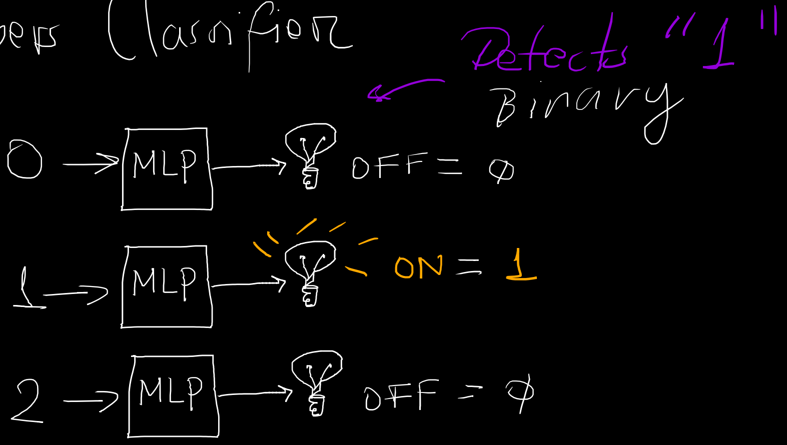
MLP

① Multi Layer Perceptron (Math model of Dense Brain)

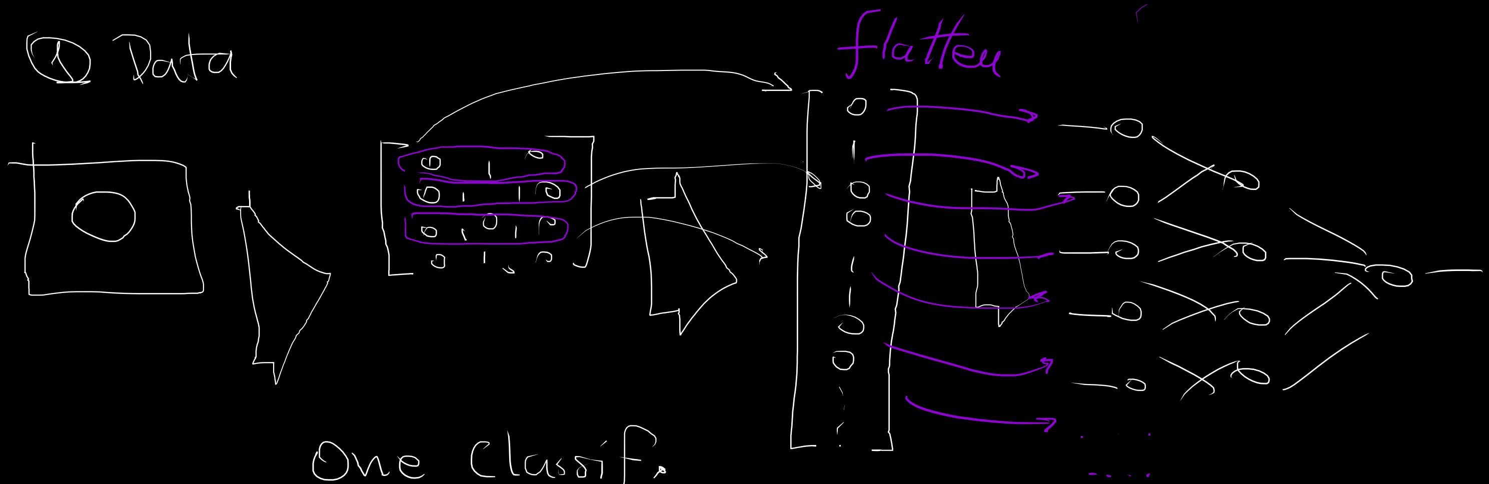


*② Practical Example: Numbers Classifion





① Data

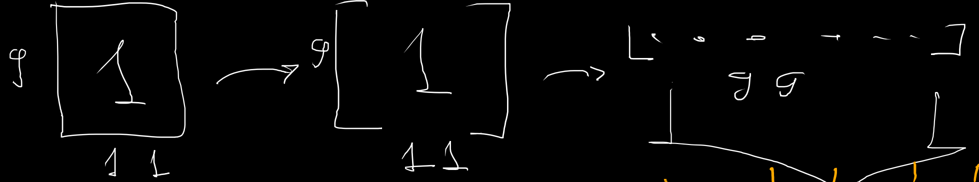


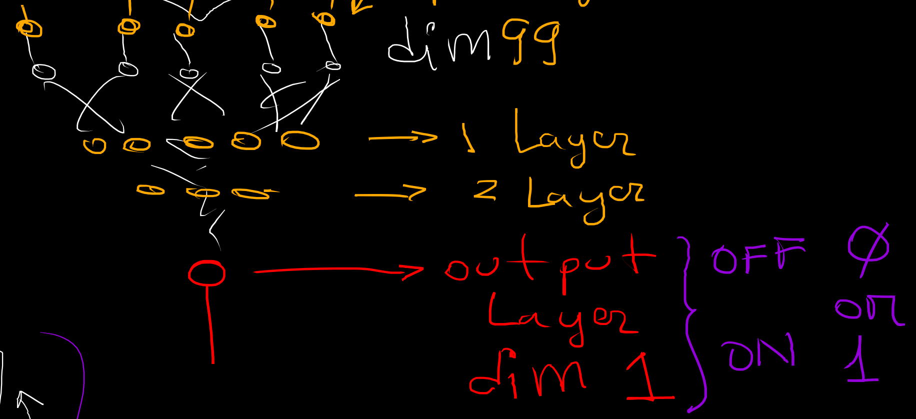
One Classif.

$\boxed{0} \rightarrow [0] \rightarrow [0] \rightarrow \text{MLP} \rightarrow \emptyset$

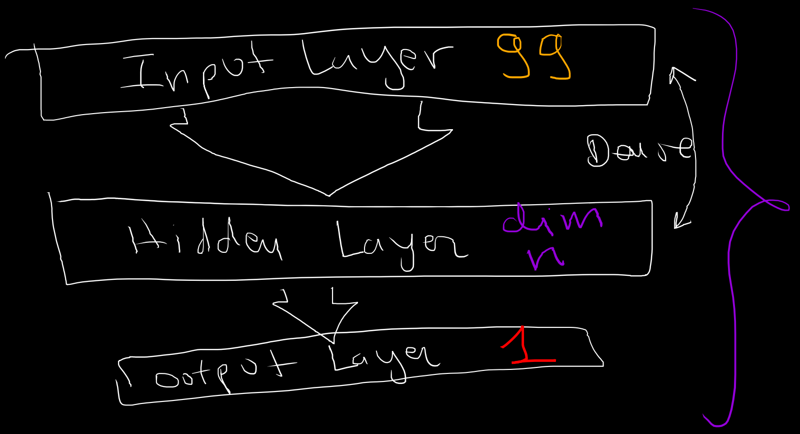
$\boxed{1} \rightarrow [1] \rightarrow [1] \rightarrow \text{MLP} \rightarrow 1$

* Implementation:





* Build Model



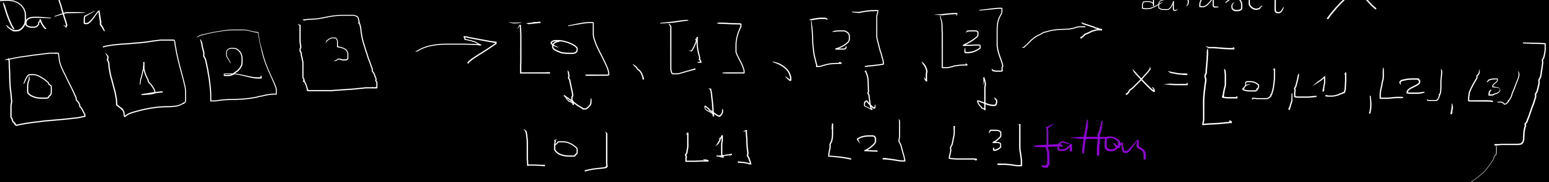
- * Model
- * Data In/out
- * Optimizer (Gradient)
- * iterative process .....

→ ∅, 1, 2, 3, ...

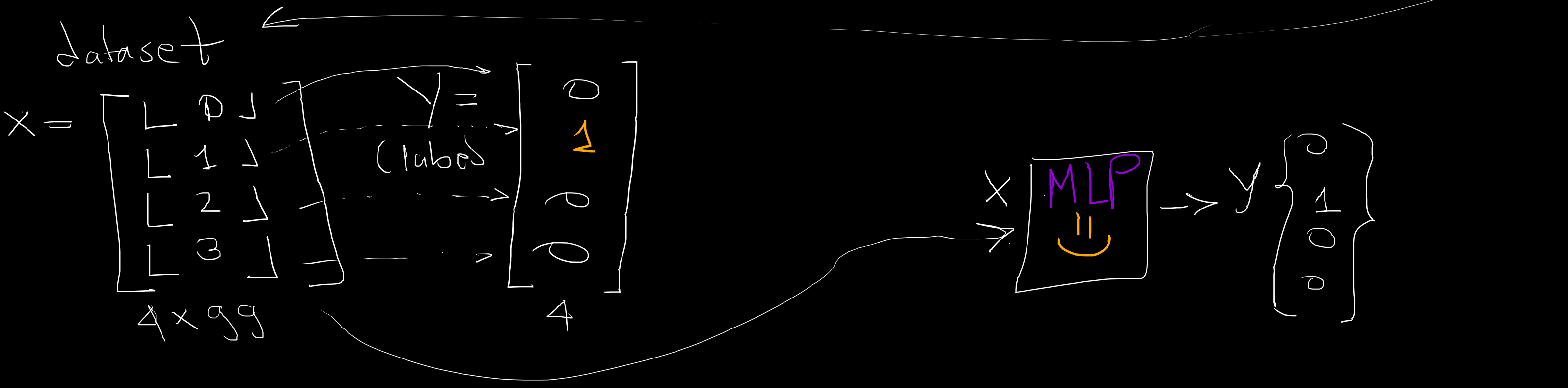
loop

* Python:

Data



0
1
2
3
[0], [1], [2], [3]
[0] [1] [2] [3] fatton
dataset X
X = [[0], [1], [2], [3]]



dataset
X = [[0] [1] [2] [3]]
4×99
Y = [0 1 0 0]
(labels)
4
X
MLP
Y
0
1
0
0