**Title:** Lecture - Machine Learning, PHYS 777

**Speakers:** Mohamed Hibat Allah

**Collection/Series:** Machine Learning (Elective), PHYS 777, February 24 - March 28, 2025

**Subject:** Condensed Matter, Other

**Date:** February 26, 2025 - 9:00 AM

**URL:** https://pirsa.org/25020016

## Lecture 2

**Today:**

→ Linear regression.
→ Parameter optimization using gradient descent.
→ Logistic regression (Classification).

**Last time:** (SL setup)

$$D = \{(\vec{x}, \vec{y})\}$$

Datapoints

$$\vec{x} = (x_1, x_2, \dots, x_{d_x})^T$$

Labels

$$\vec{y} = (y_1, y_2, \dots, y_{d_y})^T$$

**Task:** Fit some function $f(\vec{x})$ to $\vec{y}$

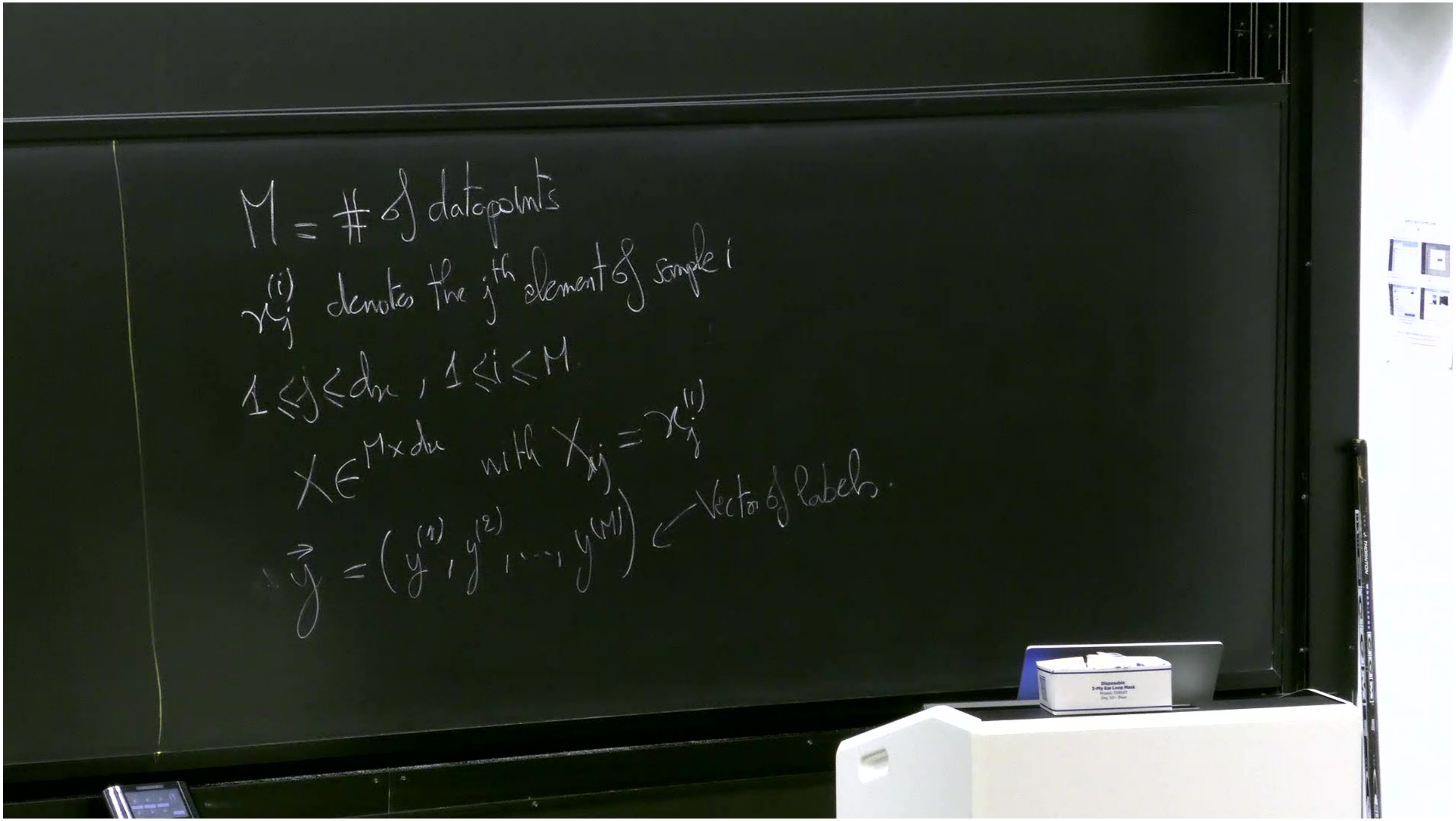
where $f : \mathbb{R}^{d_x} \longrightarrow \mathbb{R}^{d_y}$.

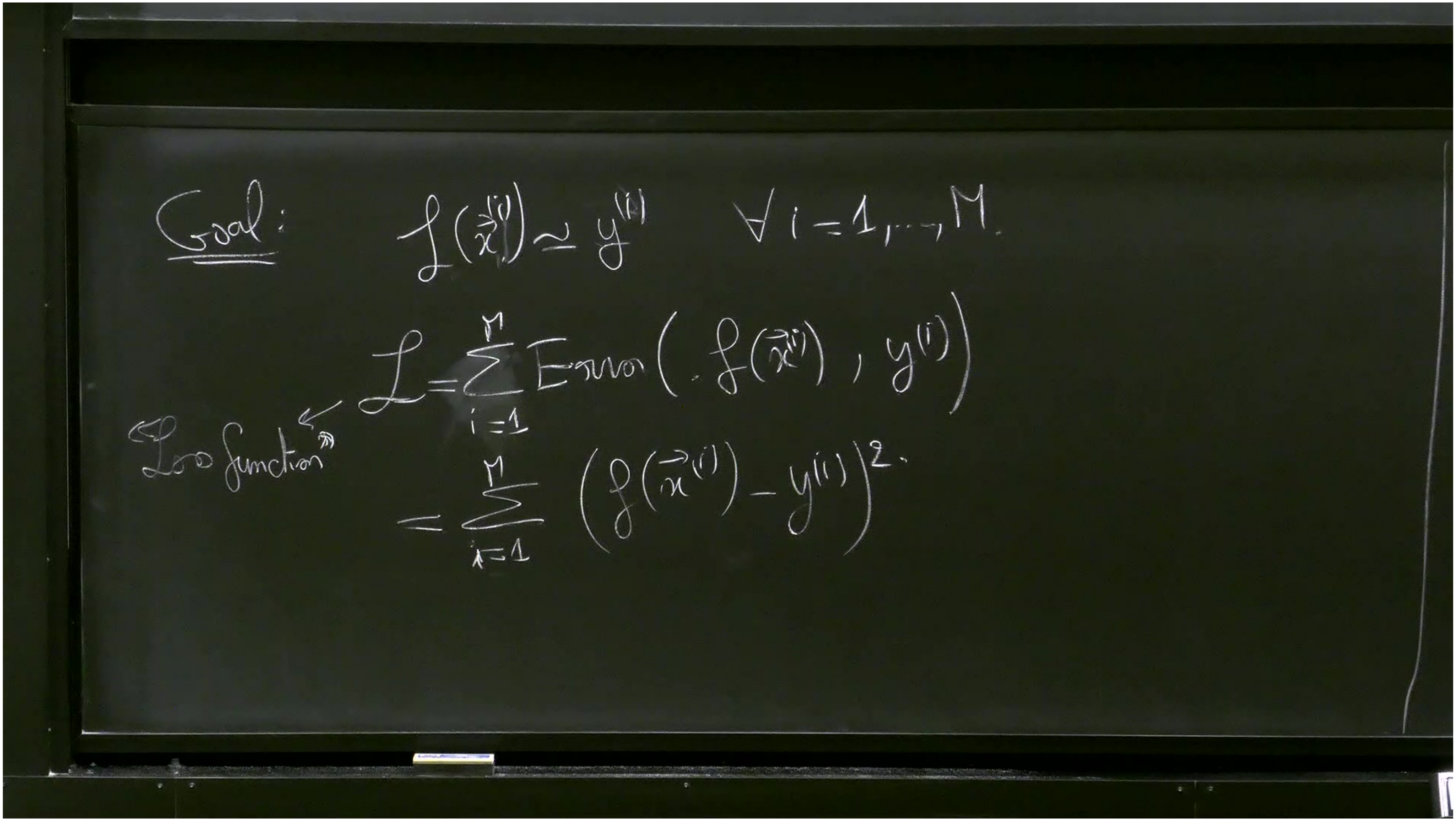
**Linear regression:** our first SL algorithm.

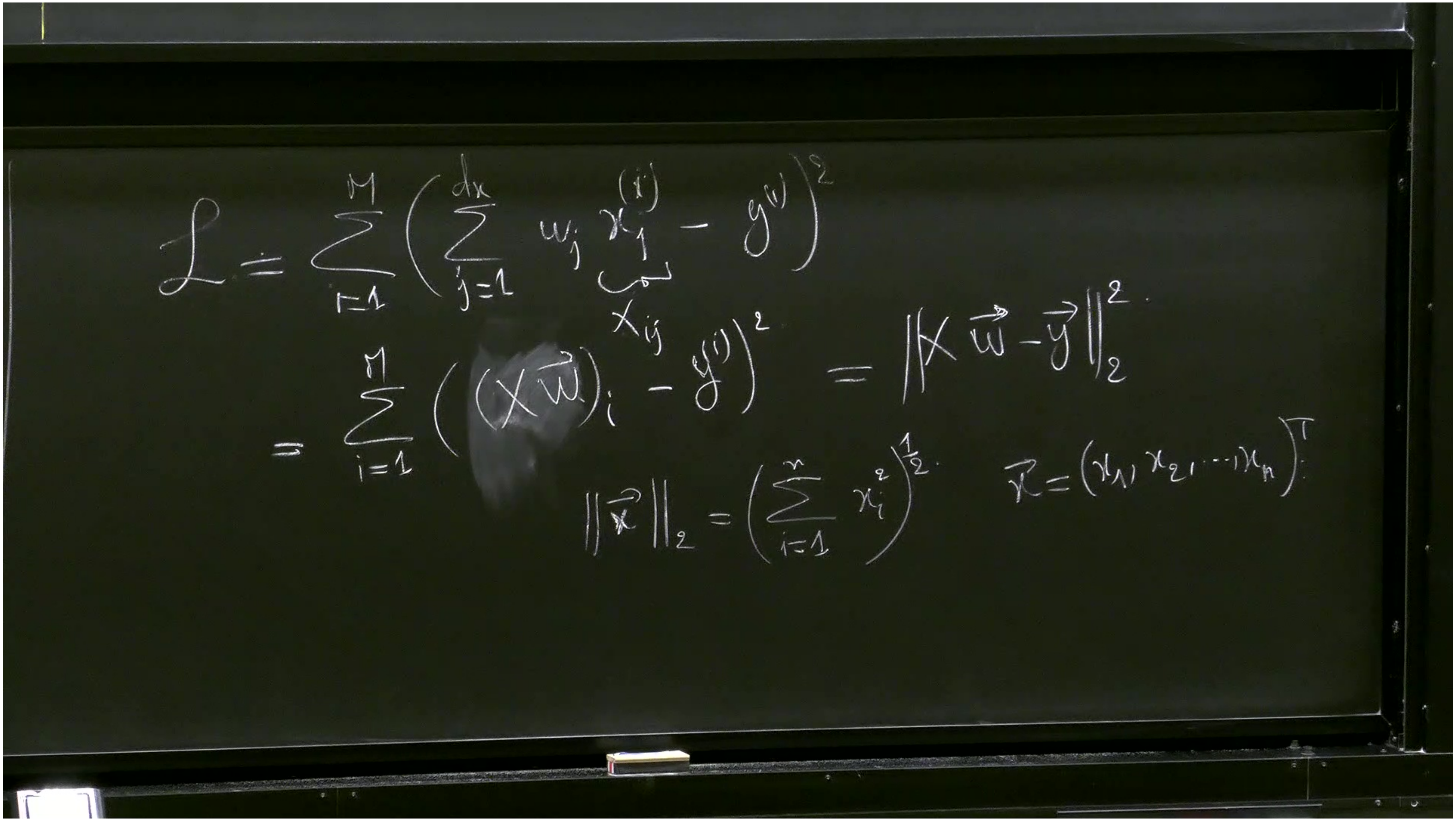
For. $D = \{(\vec{x}, y)\}$

$$f(\vec{x}) = \sum_{j=1}^{d_x} w_j x_j = \vec{w}^T \vec{x}$$

$\vec{w} \to (w_1, w_2, \dots, w_N)$.


M = # of datapoints
$x_j^{(i)}$ denotes the $j^{th}$ element of sample i
$1 \leq j \leq d_x$ , $1 \leq i \leq M$.
$X \in {}^{M \times d_x}$ with $X_{ij} = x_j^{(i)}$
$\vec{y} = (y^{(1)}, y^{(2)}, \ldots, y^{(M)})$ ← Vector of labels.


Goal: $f(\vec{x}^{(i)}) \simeq y^{(i)} \quad \forall i = 1, \dots, M.$
$\mathcal{L} = \sum_{i=1}^{M} \text{Error}\left( f(\vec{x}^{(i)}), y^{(i)} \right)$
« Loss function »
$= \sum_{i=1}^{M} \left( f(x^{(i)}) - y^{(i)} \right)^2.$

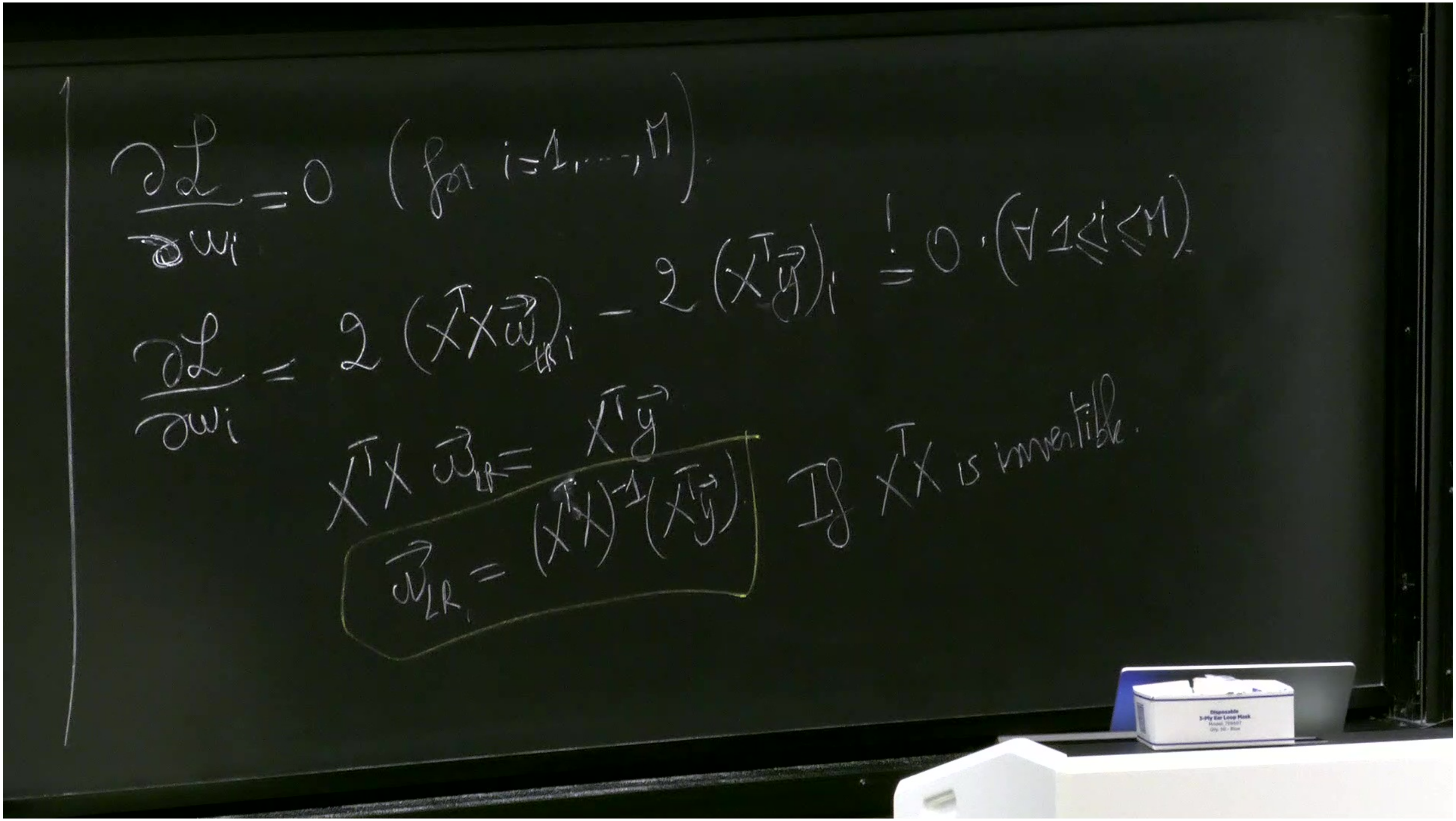

$\mathcal{L} \doteq \sum_{i=1}^{M} \Big( \sum_{j=1}^{d_x} w_j \underbrace{x_j^{(i)}}_{X_{ij}} - y^{(i)} \Big)^2$
$= \sum_{i=1}^{M} \big( (X\vec{w})_i - y^{(i)} \big)^2 = \| X\vec{w} - \vec{y} \|_2^2$
$\|\vec{x}\|_2 = \Big( \sum_{i=1}^{n} x_i^2 \Big)^{\frac{1}{2}}$
$\vec{x} = (x_1, x_2, \dots, x_n)^T$

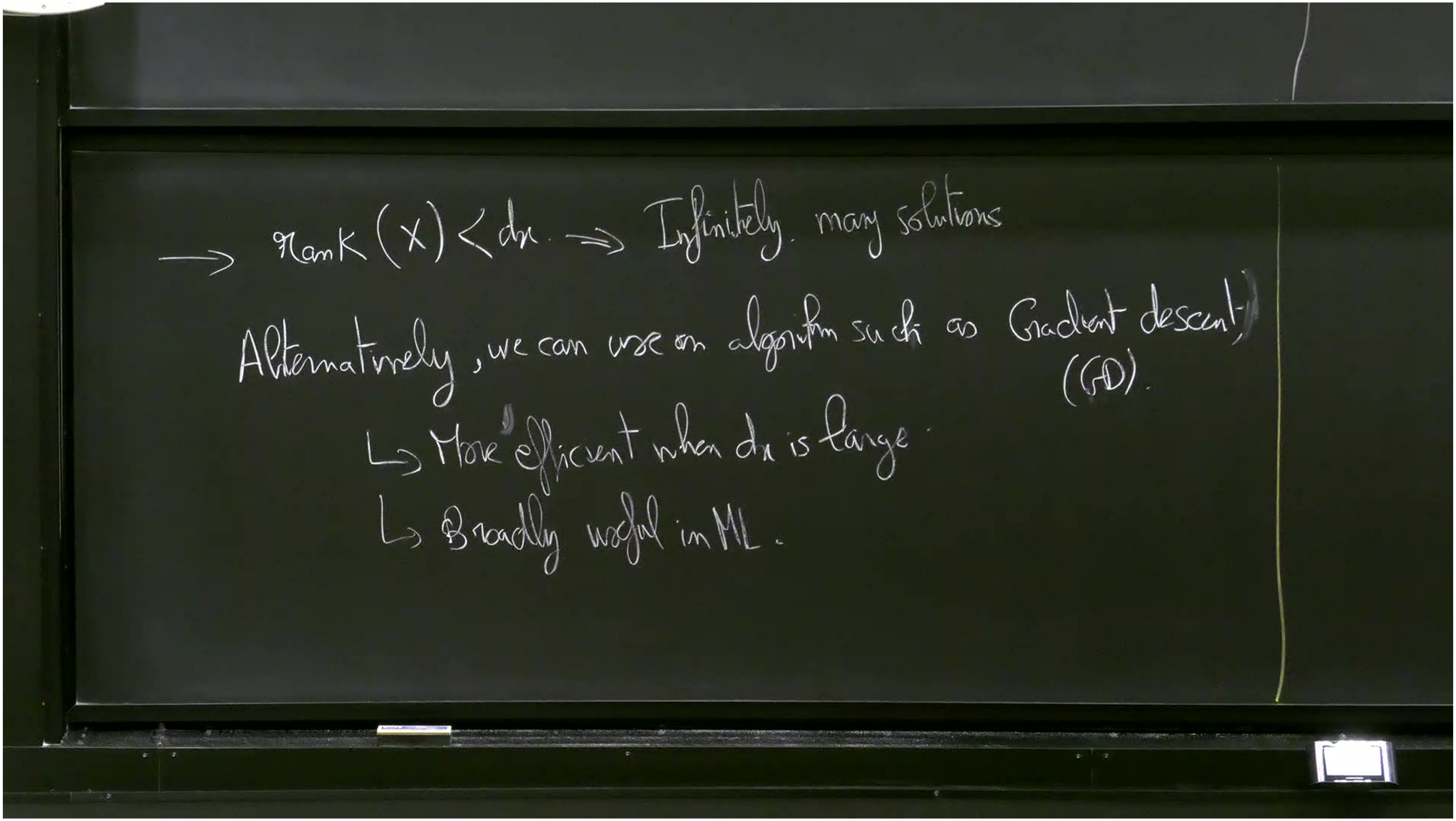
Let's solve the equation $\vec{w}_{LR} = \underset{\vec{w}\in\mathbb{R}^{d_x}}{\arg\min}\ \mathcal{L}$.

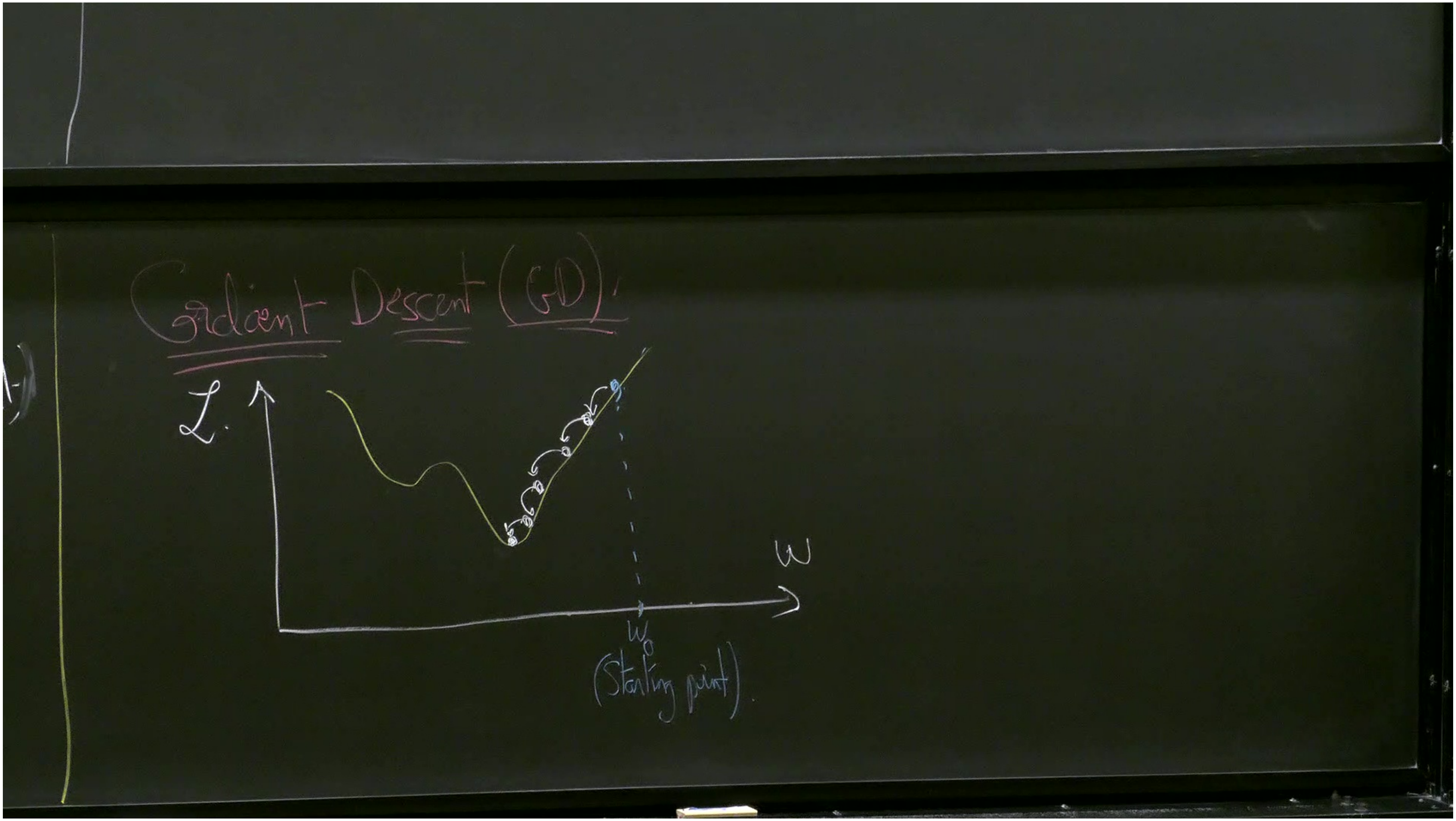
$$\mathcal{L} = (X\vec{w}-\vec{y})^T(X\vec{w}-\vec{y})$$

$$= \vec{w}^T X^T X \vec{w} + y^T y - w^T X^T \vec{y} - \vec{y}^T X \vec{w}$$

$$=$$


$\frac{\partial \mathcal{L}}{\partial w_i} = 0$ (for $i = 1, \dots, M$).
$\frac{\partial \mathcal{L}}{\partial w_i} = 2(X^T X \vec{w}_{LR})_i - 2(X^T \vec{y})_i \stackrel{!}{=} 0 \cdot (\forall\, 1 \leq i \leq M)$.
$X^T X \vec{w}_{LR} = X^T \vec{y}$
$\vec{w}_{LR} = (X^T X)^{-1}(X^T \vec{y})$.
If $X^T X$ is invertible.


→ rank(X) < dx ⇒ Infinitely many solutions
Alternatively, we can use an algorithm such as Gradient descent (GD).
↳ More efficient when dx is large.
↳ Broadly useful in ML.


Gradient Descent (GD):
L.
w
w0
(Starting point).

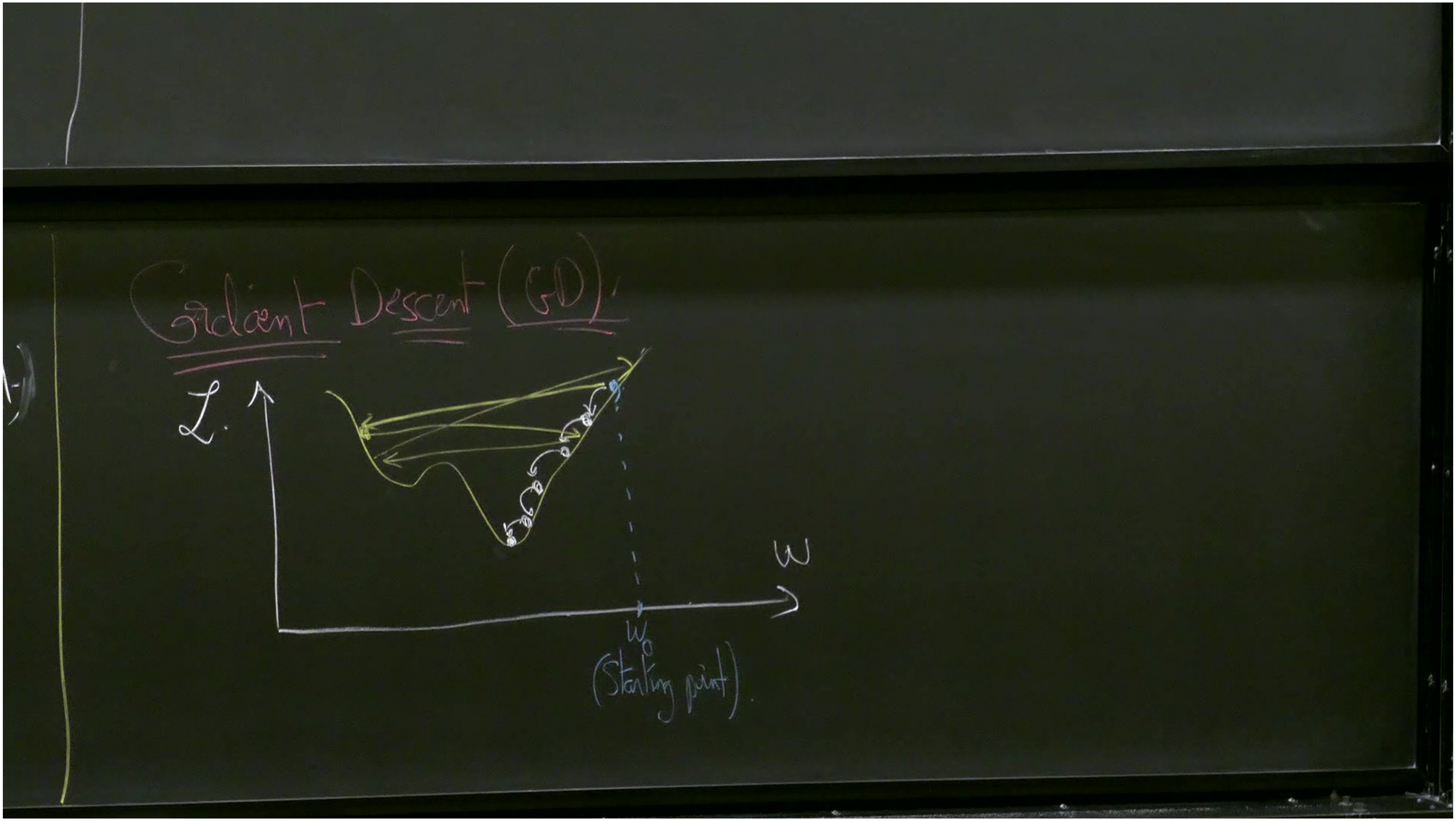
$$\Delta \mathcal{L} \simeq \sum_{i=1}^{n} \frac{\partial \mathcal{L}}{\partial w_i} \Delta w_i = -\eta \sum_{i=1}^{n} \left(\frac{\partial \mathcal{L}}{\partial w_i}\right)^2 \qquad \vec{w} = (w_1, w_2, \dots, w_n)$$

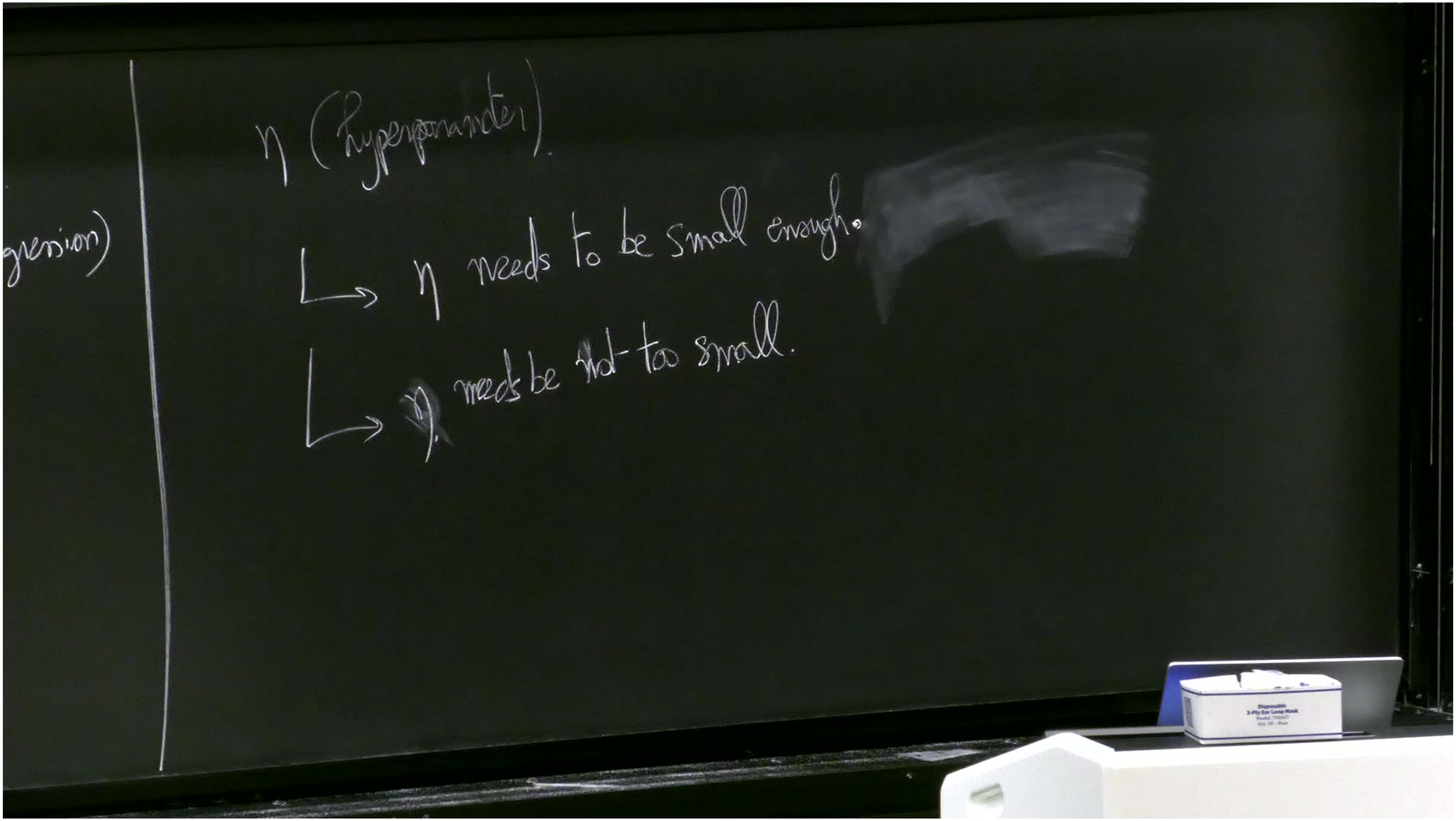
$(n = d_x$ for Linear regression$)$

$$\Delta \mathcal{L} \overset{(1)}{\leq} 0$$

$$\Delta w_i = -\eta \frac{\partial \mathcal{L}}{\partial w_i}$$

$\eta > 0$ "Learning rate"


Gradient Descent (GD):
L.
w
w0
(Starting point).


η (hyperparameter).
gression)
↳ η needs to be small enough.
↳ η needs be not too small.

## Logistic regression (Classification)

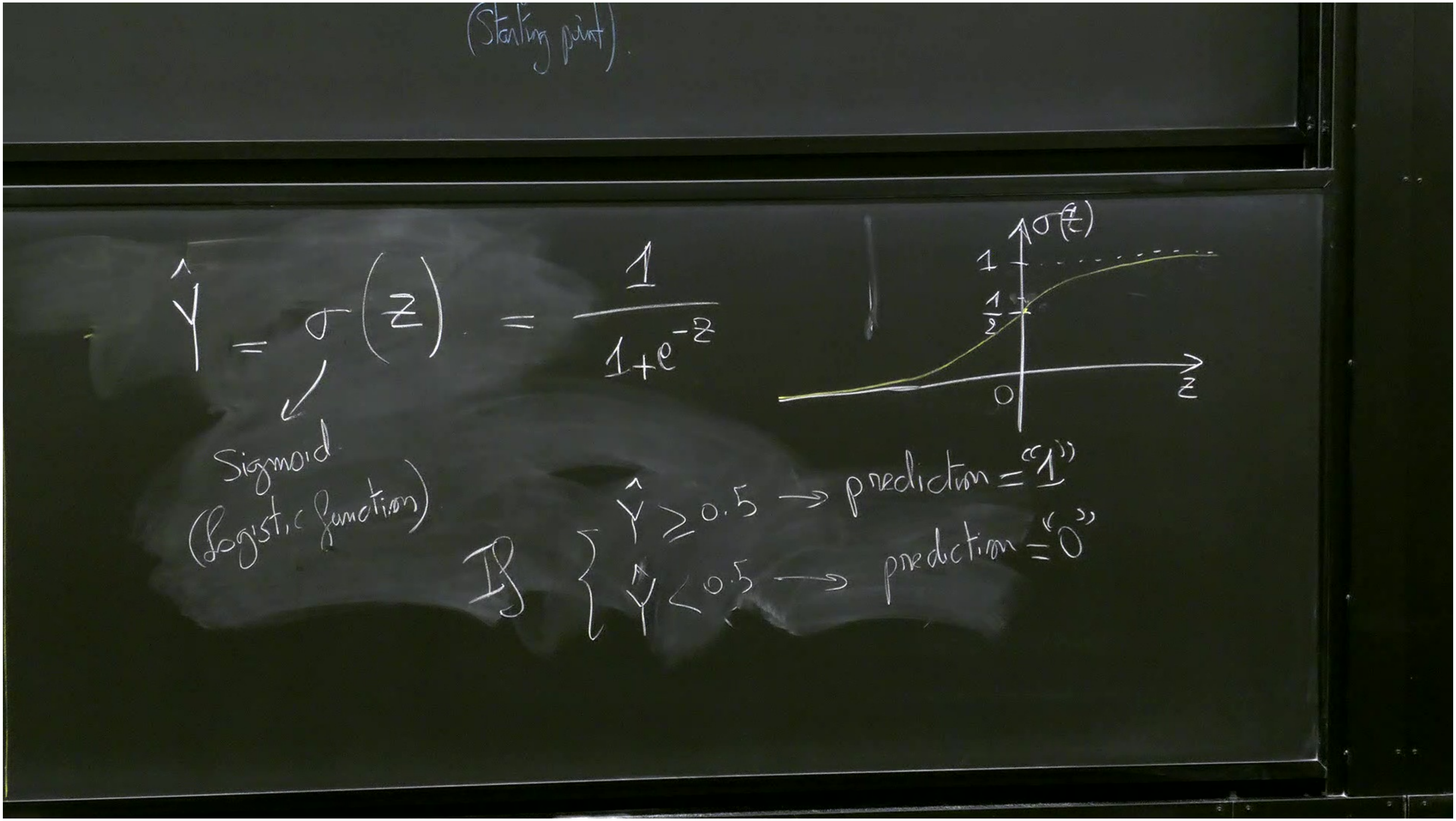
↳ More suitable for predicting binary classes "0, 1"

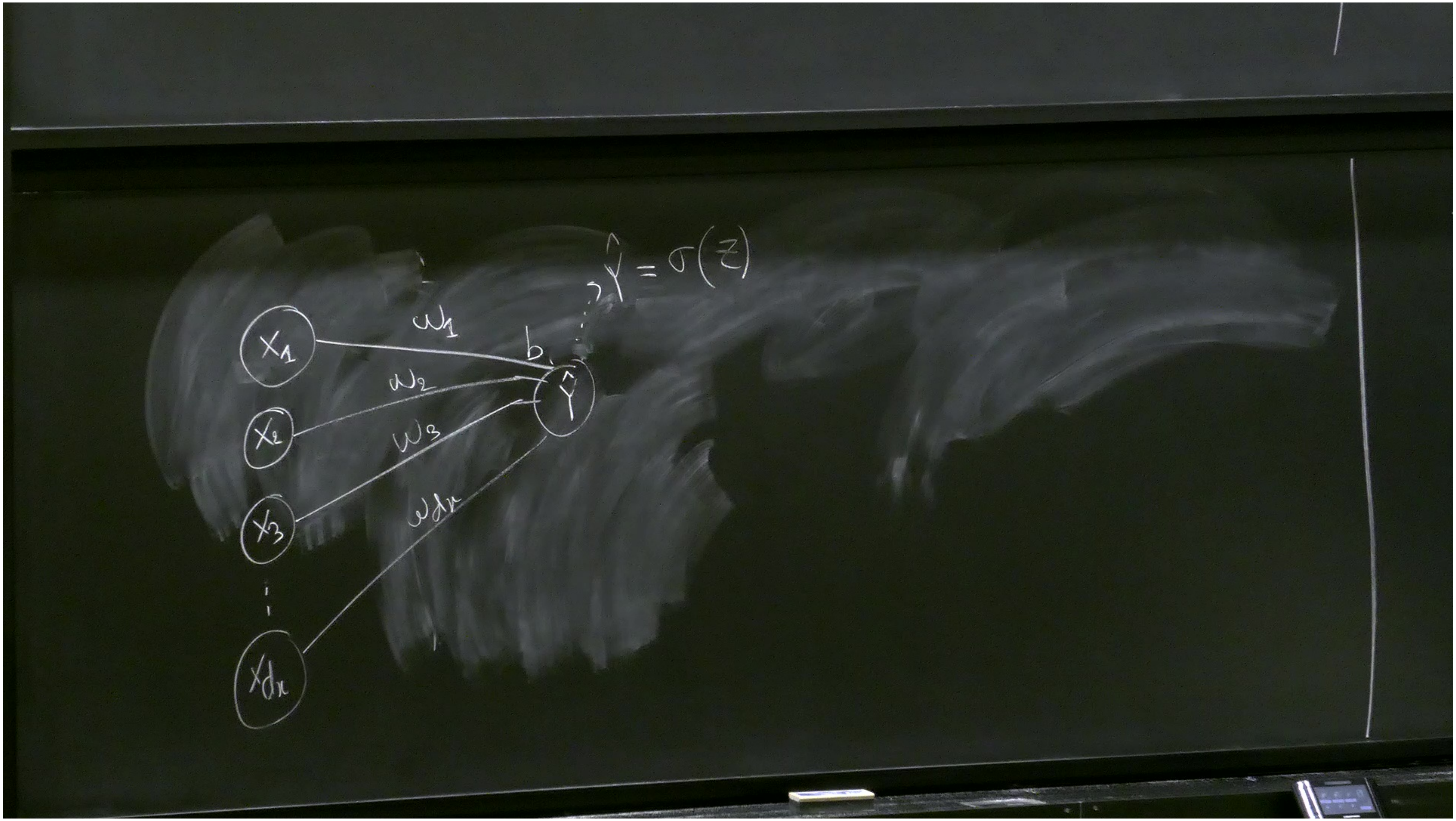
↳ Given $\vec{x} = (x_1, x_2, \cdots, x_{d_x})$.

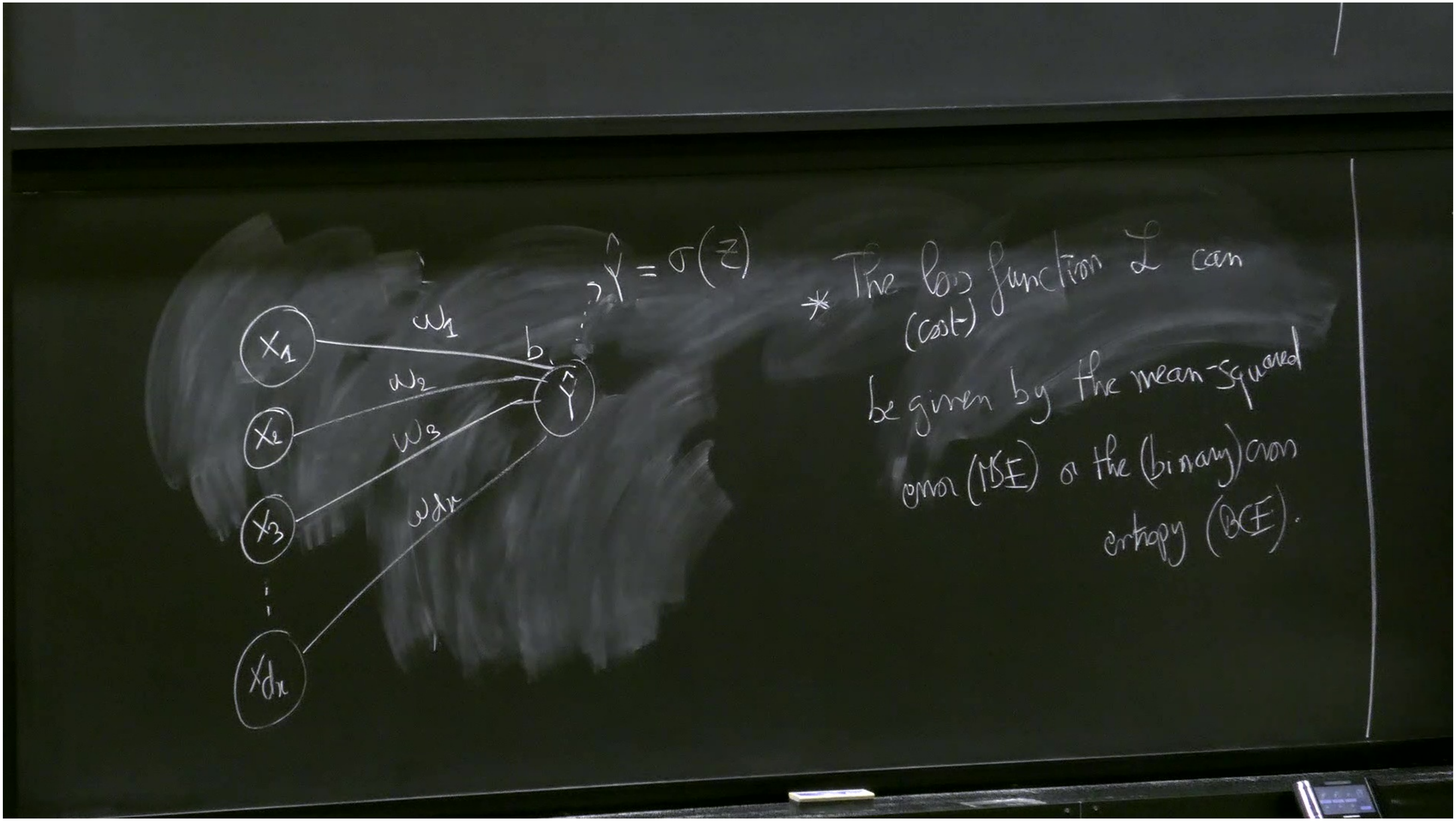
$$z = w_1 x_1 + w_2 x_2 + \cdots + w_{d_x} x_{d_x} + b$$

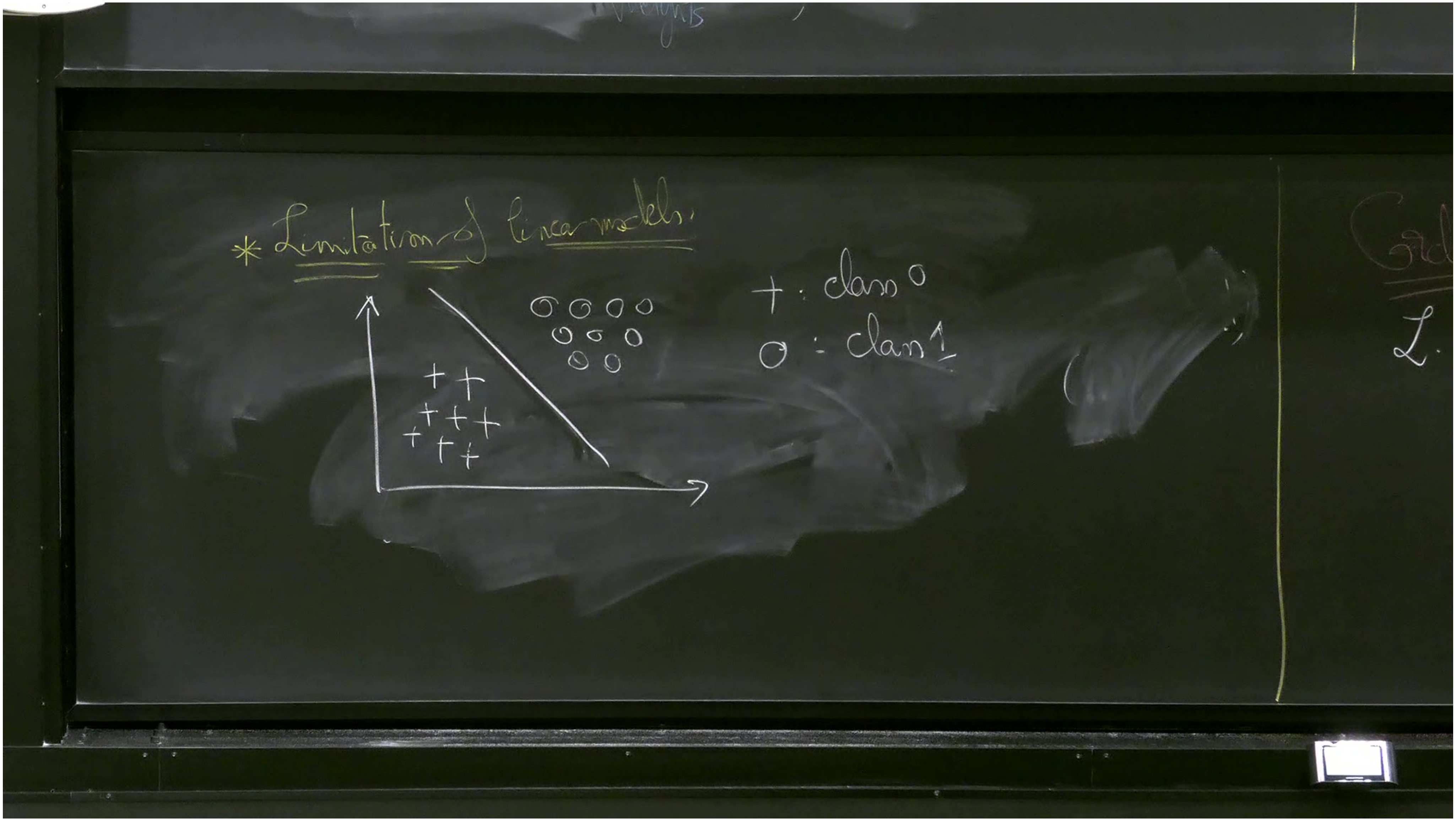
weights (→ $w_1, w_2, \ldots, w_{d_x}$)

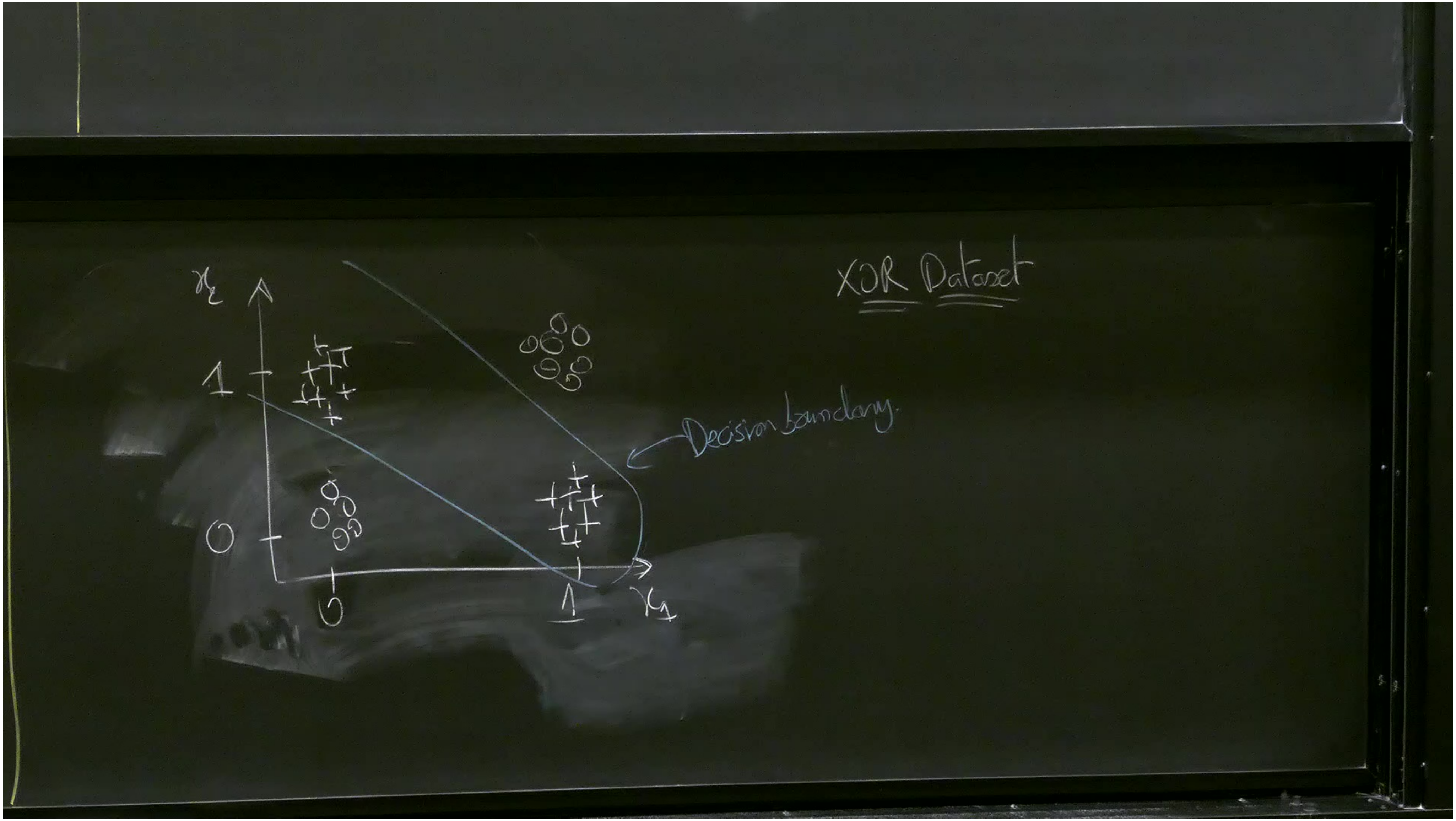
bias (→ $b$)

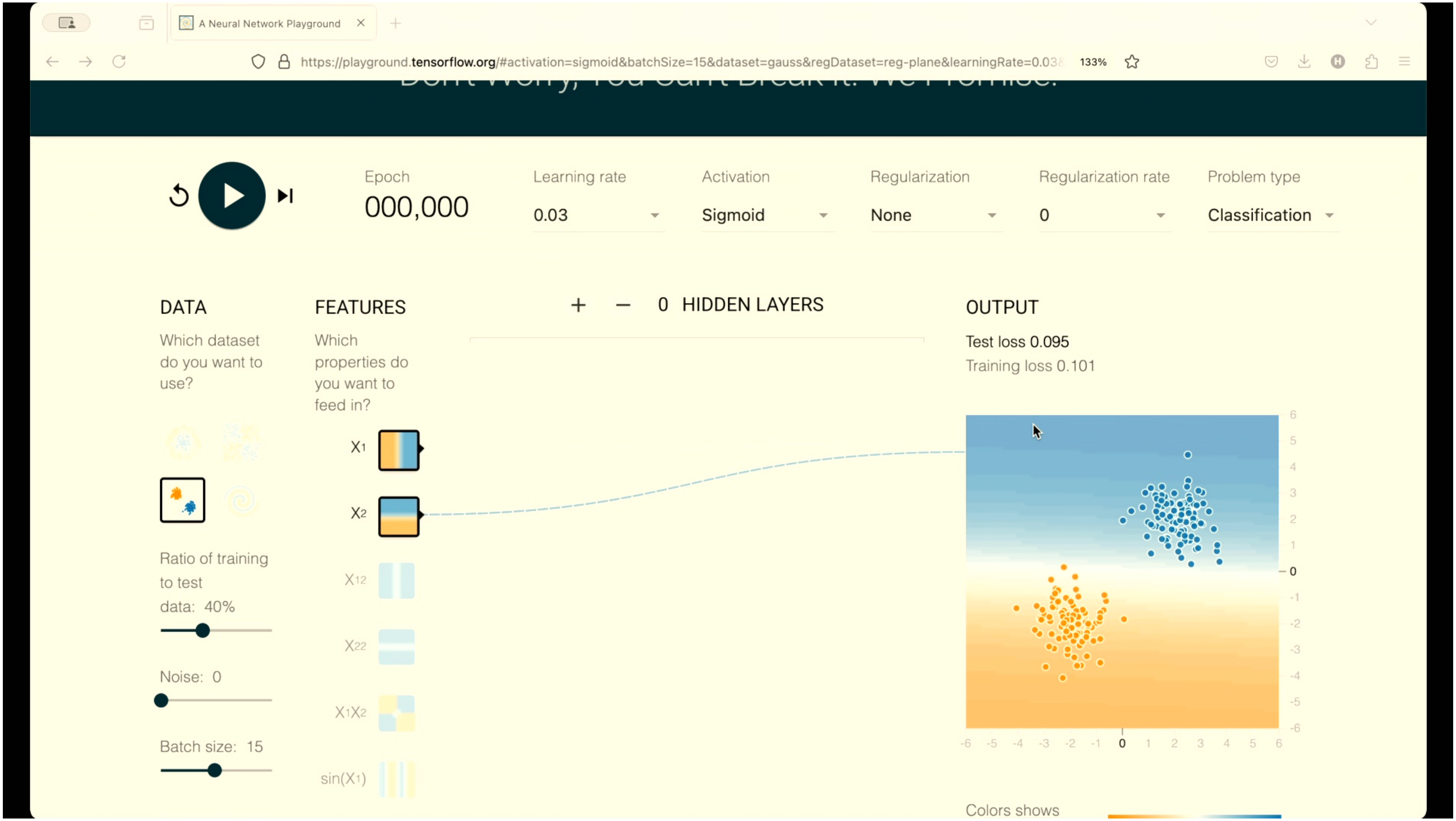

(Starting point).
$\hat{Y} = \sigma(z) = \frac{1}{1+e^{-z}}$
Sigmoid
(Logistic function)
$\sigma(z)$
1
$\frac{1}{2}$
0
z
If $\begin{cases} \hat{Y} \geq 0.5 \rightarrow \text{prediction} = \text{"1"} \\ \hat{Y} < 0.5 \rightarrow \text{prediction} = \text{"0"} \end{cases}$


$\hat{y} = \sigma(z)$
$x_1$
$x_2$
$x_3$
$x_{d_x}$
$w_1$
$w_2$
$w_3$
$w_{d_x}$
b
$\hat{y}$

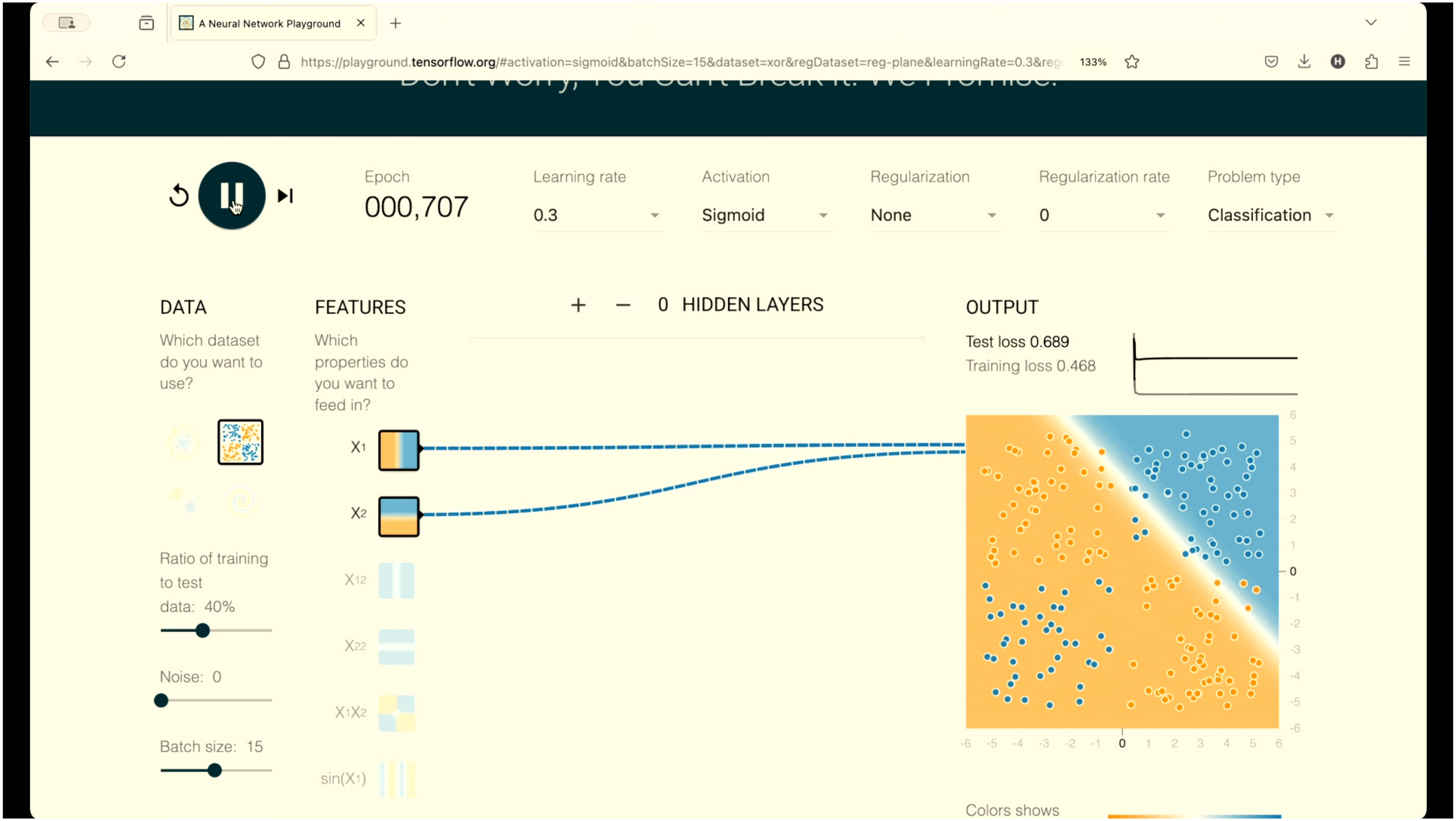

ŷ = σ(z)
x1
x2
x3
xdx
w1
w2
w3
wdx
b
ŷ
* The loss function ℒ can
(cost)
be given by the mean-squared
error (MSE) or the (binary) cross
entropy (BCE).

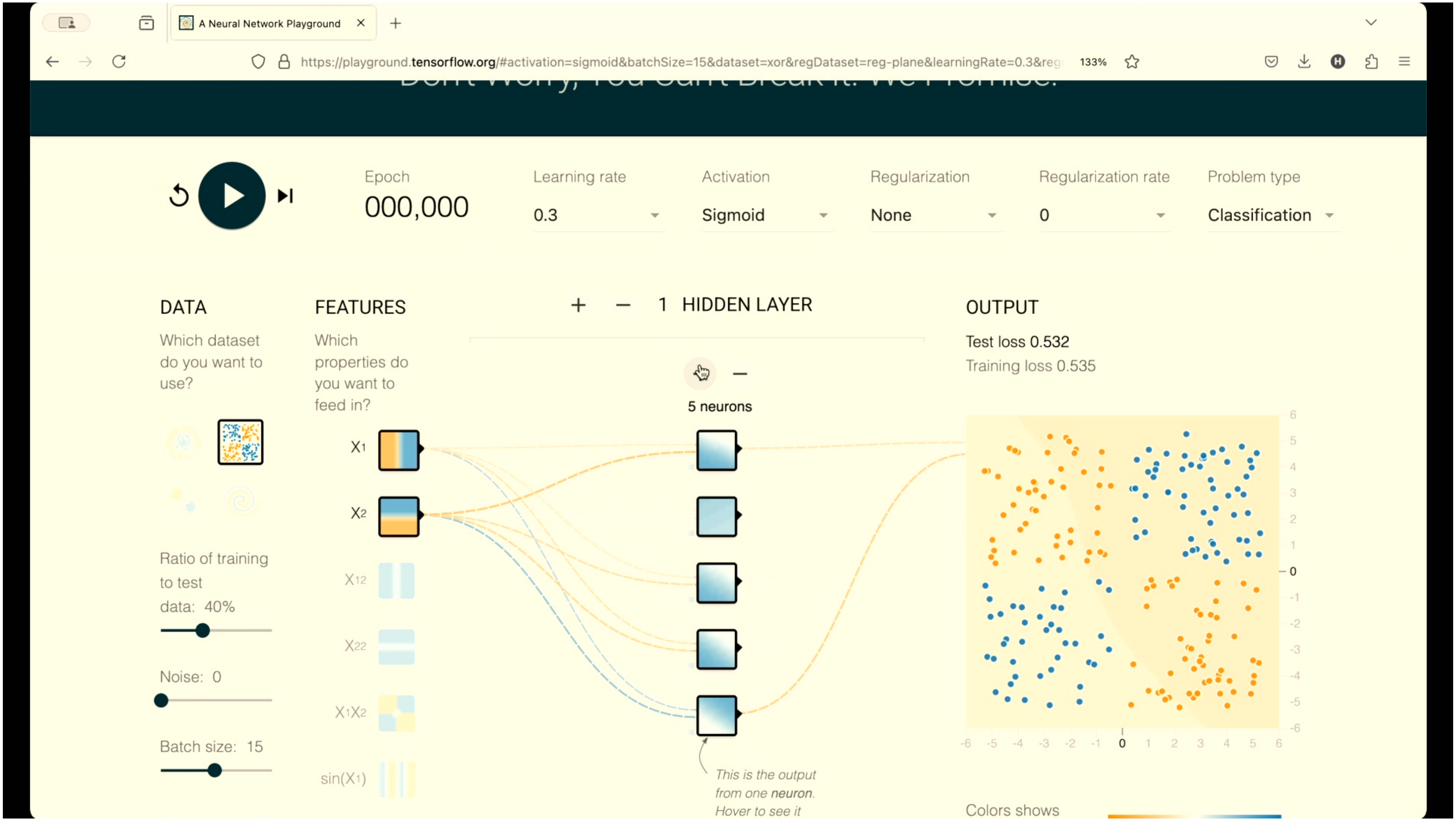

* Limitations of linear models:
+ : class 0
O : class 1

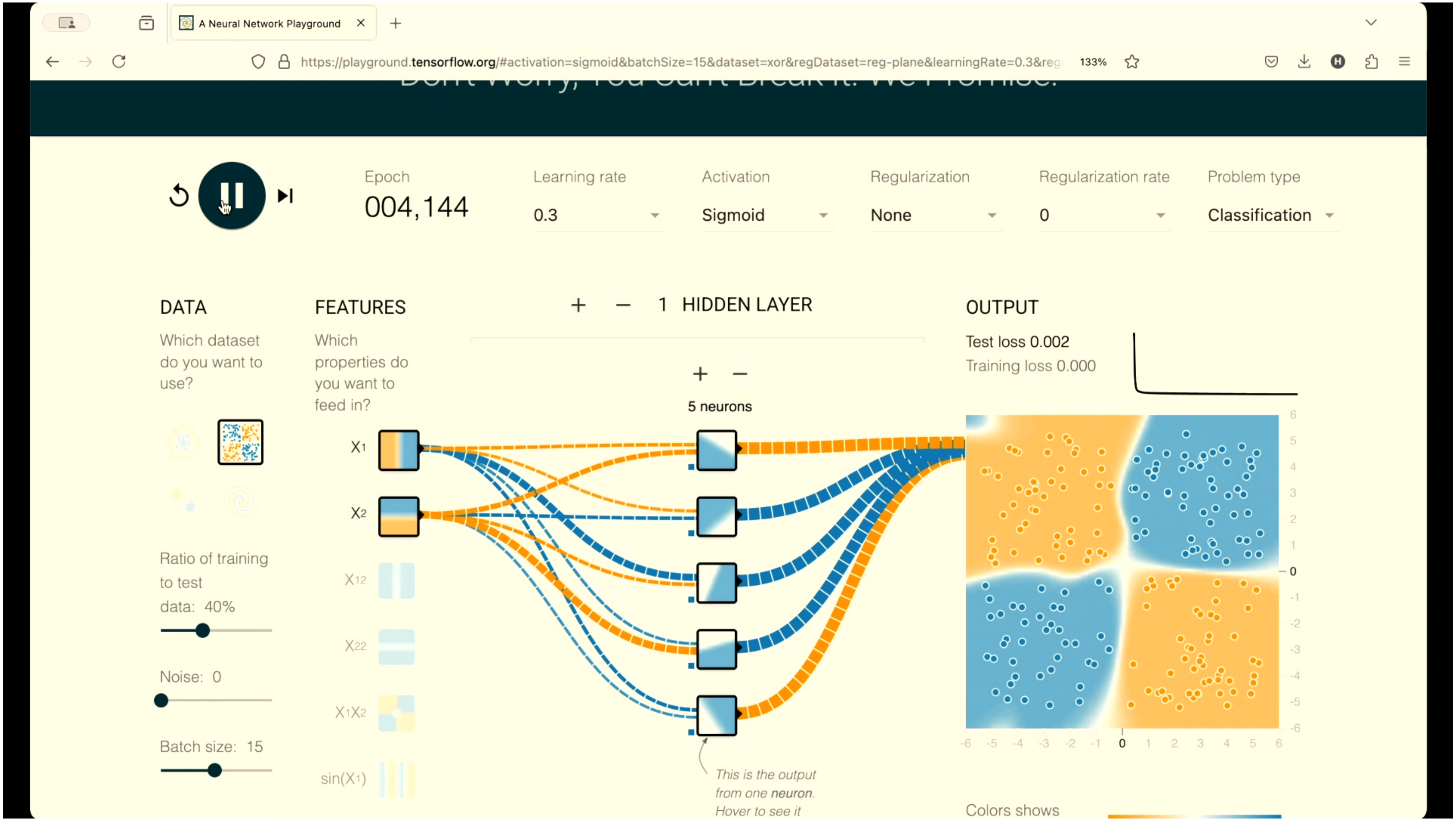

XOR Dataset
Decision boundary.
$x_2$
$x_1$
1
0
0
1

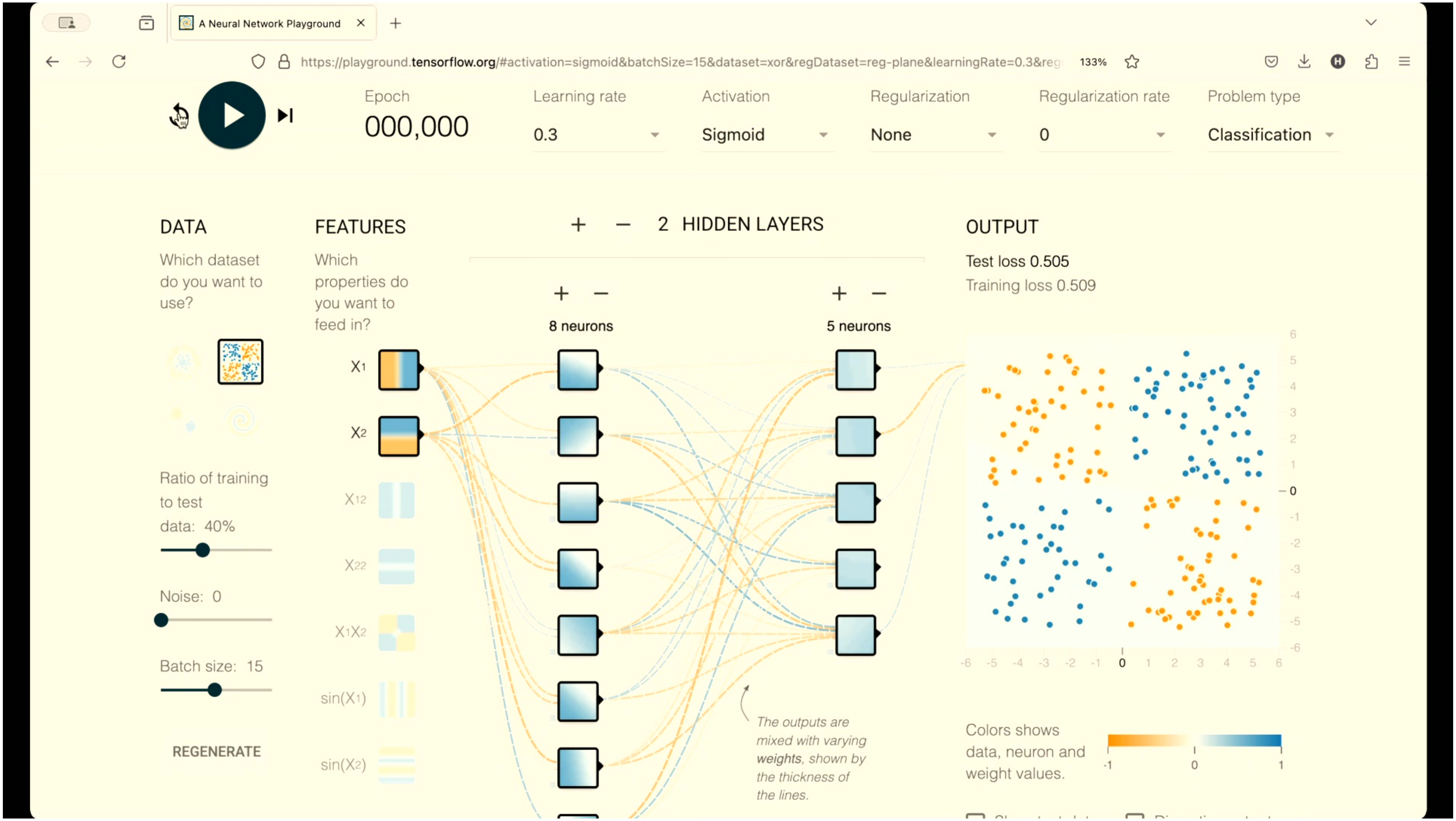


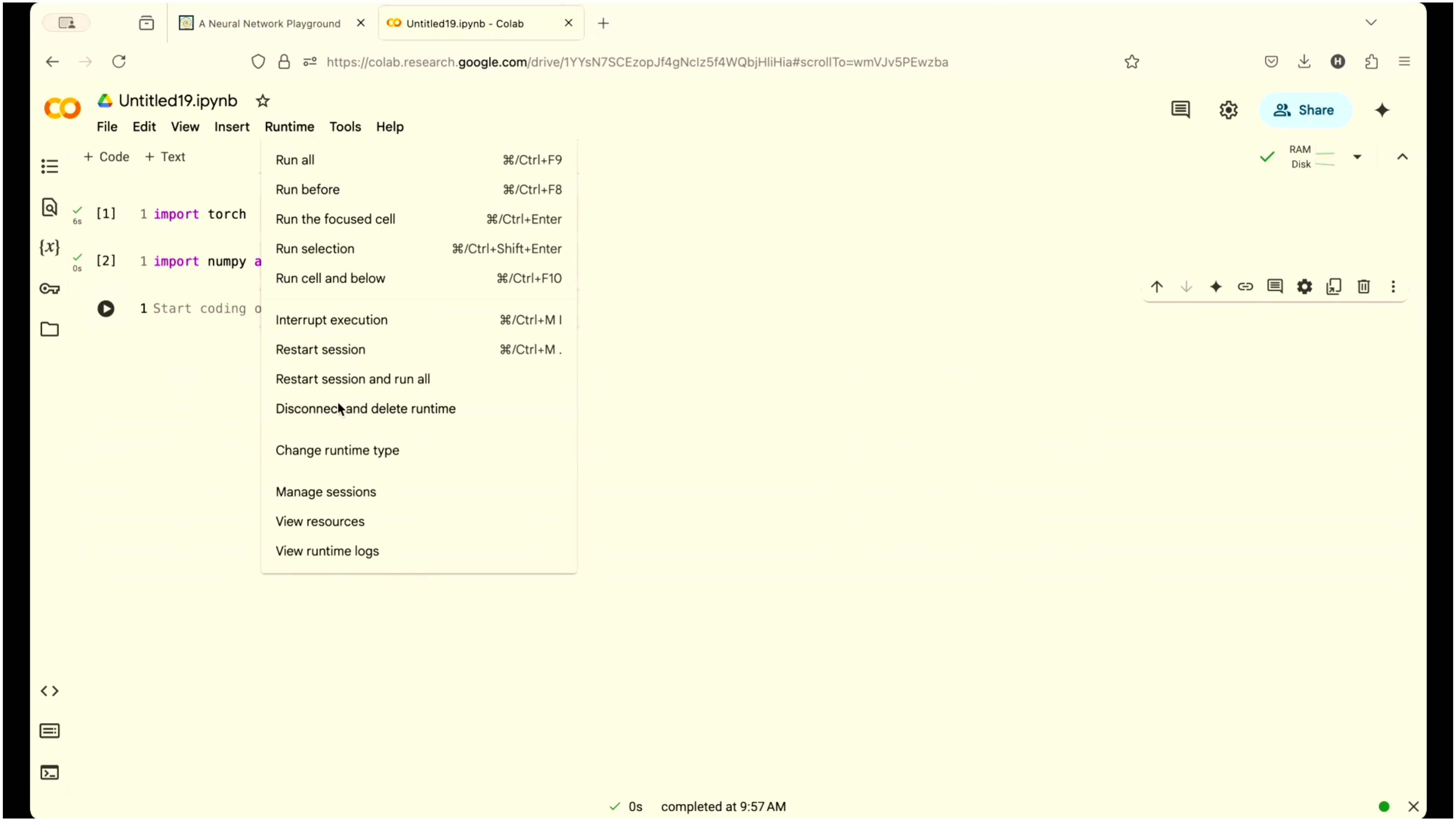
A Neural Network Playground
https://playground.tensorflow.org/#activation=sigmoid&batchSize=15&dataset=gauss&regDataset=reg-plane&learningRate=0.03
133%
Epoch
000,000
Learning rate
0.03
Activation
Sigmoid
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 40%
Noise: 0
Batch size: 15
FEATURES
Which properties do you want to feed in?
X1
X2
+ − 0 HIDDEN LAYERS
OUTPUT
Test loss 0.095
Training loss 0.101
Colors shows


A Neural Network Playground
https://playground.tensorflow.org/#activation=sigmoid&batchSize=15&dataset=xor&regDataset=reg-plane&learningRate=0.3&reg
133%
Epoch
000,707
Learning rate
0.3
Activation
Sigmoid
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
FEATURES
Which properties do you want to feed in?
+ – 0 HIDDEN LAYERS
OUTPUT
Test loss 0.689
Training loss 0.468
X1
X2
X12
X22
X1X2
sin(X1)
Ratio of training to test data: 40%
Noise: 0
Batch size: 15
Colors shows


A Neural Network Playground
https://playground.tensorflow.org/#activation=sigmoid&batchSize=15&dataset=xor&regDataset=reg-plane&learningRate=0.3&reg
133%
Epoch
000,000
Learning rate
0.3
Activation
Sigmoid
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 40%
Noise: 0
Batch size: 15
FEATURES
Which properties do you want to feed in?
1 HIDDEN LAYER
5 neurons
This is the output from one neuron. Hover to see it
OUTPUT
Test loss 0.532
Training loss 0.535
Colors shows


A Neural Network Playground
https://playground.tensorflow.org/#activation=sigmoid&batchSize=15&dataset=xor&regDataset=reg-plane&learningRate=0.3&reg
133%
Epoch
004,144
Learning rate
0.3
Activation
Sigmoid
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
FEATURES
Which properties do you want to feed in?
1 HIDDEN LAYER
5 neurons
OUTPUT
Test loss 0.002
Training loss 0.000
Ratio of training to test data: 40%
Noise: 0
Batch size: 15
This is the output from one neuron. Hover to see it
Colors shows


A Neural Network Playground
https://playground.tensorflow.org/#activation=sigmoid&batchSize=15&dataset=xor&regDataset=reg-plane&learningRate=0.3&reg
133%
Epoch
000,000
Learning rate
0.3
Activation
Sigmoid
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 40%
Noise: 0
Batch size: 15
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
2 HIDDEN LAYERS
8 neurons
5 neurons
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.505
Training loss 0.509
Colors shows data, neuron and weight values.


A Neural Network Playground
Untitled19.ipynb - Colab
https://colab.research.google.com/drive/1YYsN7SCEzopJf4gNclz5f4WQbjHliHia#scrollTo=wmVJv5PEwzba
Untitled19.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text
RAM
Disk
[1] 1 import torch
[2] 1 import numpy a
1 Start coding o
Run all ⌘/Ctrl+F9
Run before ⌘/Ctrl+F8
Run the focused cell ⌘/Ctrl+Enter
Run selection ⌘/Ctrl+Shift+Enter
Run cell and below ⌘/Ctrl+F10
Interrupt execution ⌘/Ctrl+M I
Restart session ⌘/Ctrl+M .
Restart session and run all
Disconnect and delete runtime
Change runtime type
Manage sessions
View resources
View runtime logs
0s completed at 9:57 AM