Title: Lecture - Machine Learning, PHYS 777

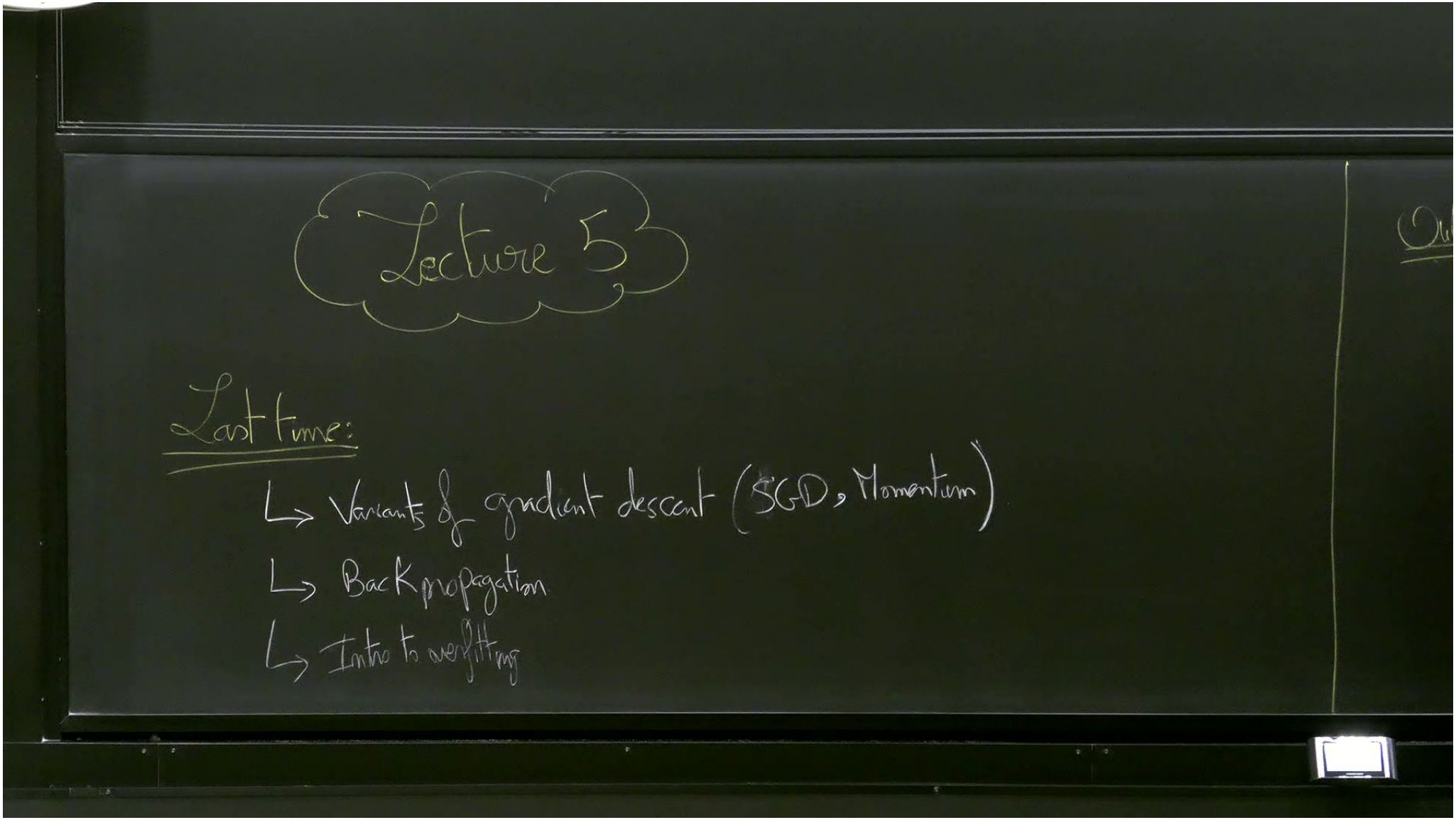
Speakers: Mohamed Hibat Allah

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

Subject: Condensed Matter, Other

Date: March 05, 2025 - 9:00 AM

URL: https://pirsa.org/25030036


Lecture 5
Last time:
↳ Variants of gradient descent (SGD, Momentum)
↳ Backpropagation
↳ Intro to overfitting

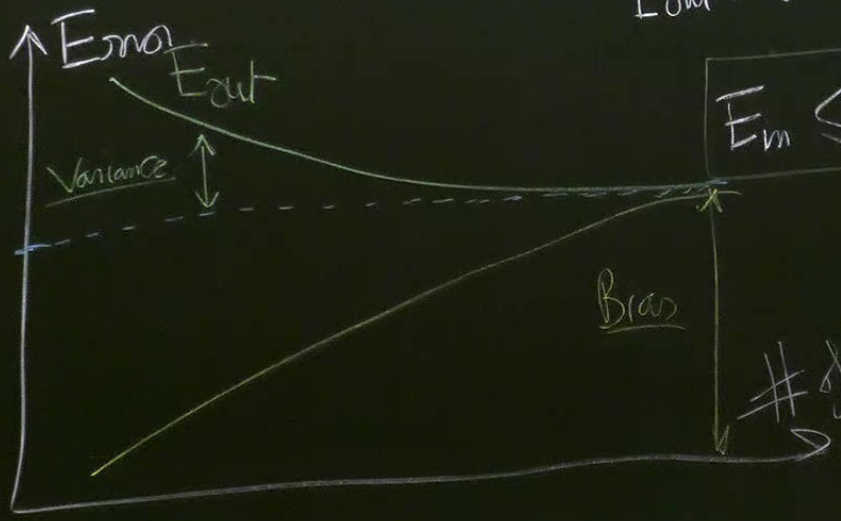
Bias-variance tradeoff

E_{in}: error on the "in-samples" (training data)

E_{out}: error on the "out-samples" (not used for training)

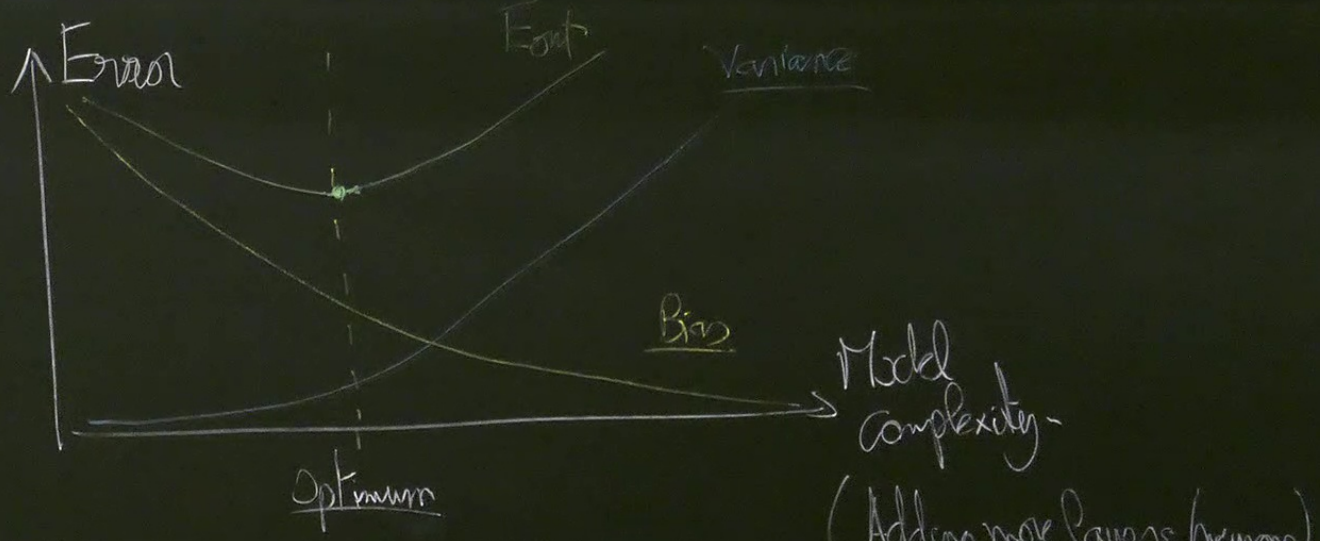
$E_{in} \leq E_{out}$

$D = \{(\vec{x}, \vec{y})\}$ → "in", "out"



For a fixed model

↳ Intro to overfitting



→ Bias. measures how well the model can do in the infinite-data limit.

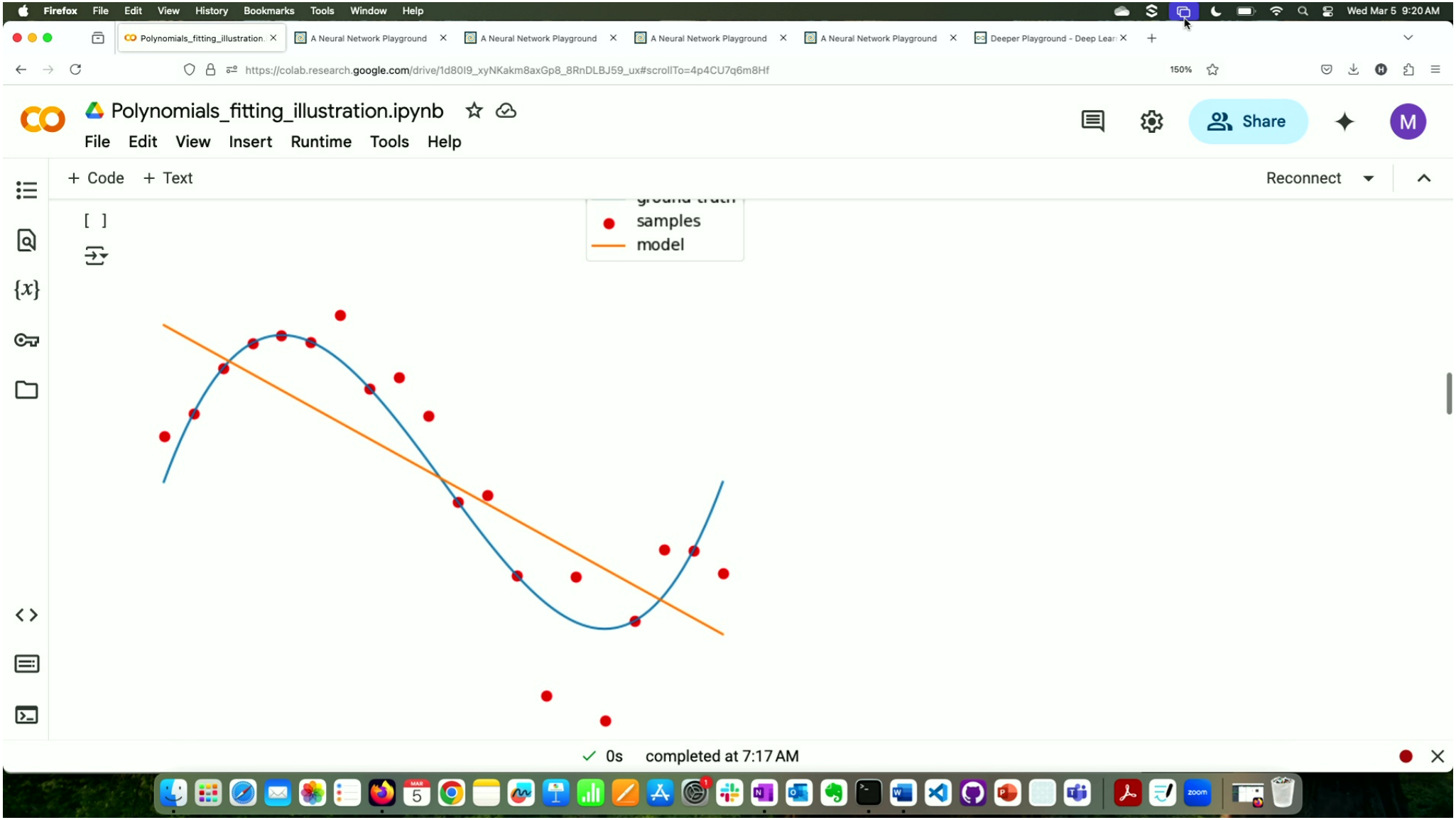
↳ More complex models tend to lower the bias

→ Variance. measures how dependent the fit is on the particular realization of the training data.

→ The difference between E_{in} and E_{out} measures the difference between «fitting» and «predicting».

↳ If $E_{out} - E_{in} \gg 1$, the model is overfitting.

neurons).


Polynomials_fitting_illustration.ipynb
File Edit View Insert Runtime Tools Help
+ Code + Text
Reconnect
[ ]
samples
model
0s completed at 7:17 AM

Polynomials_fitting_illustration.ipynb

File Edit View Insert Runtime Tools Help

Share

+ Code + Text

Reconnect

The following notebook plots the result of using gradient descent to learn a degree 'd' Legendre polynomial on training samples. The default configuration uses training samples which are essentially a cubic with noise (feel free to change this below).

Note that the output of gradient descent starting from 0 initialization of the objective

$$\min_{\beta} ||X\beta - y||_2^2$$

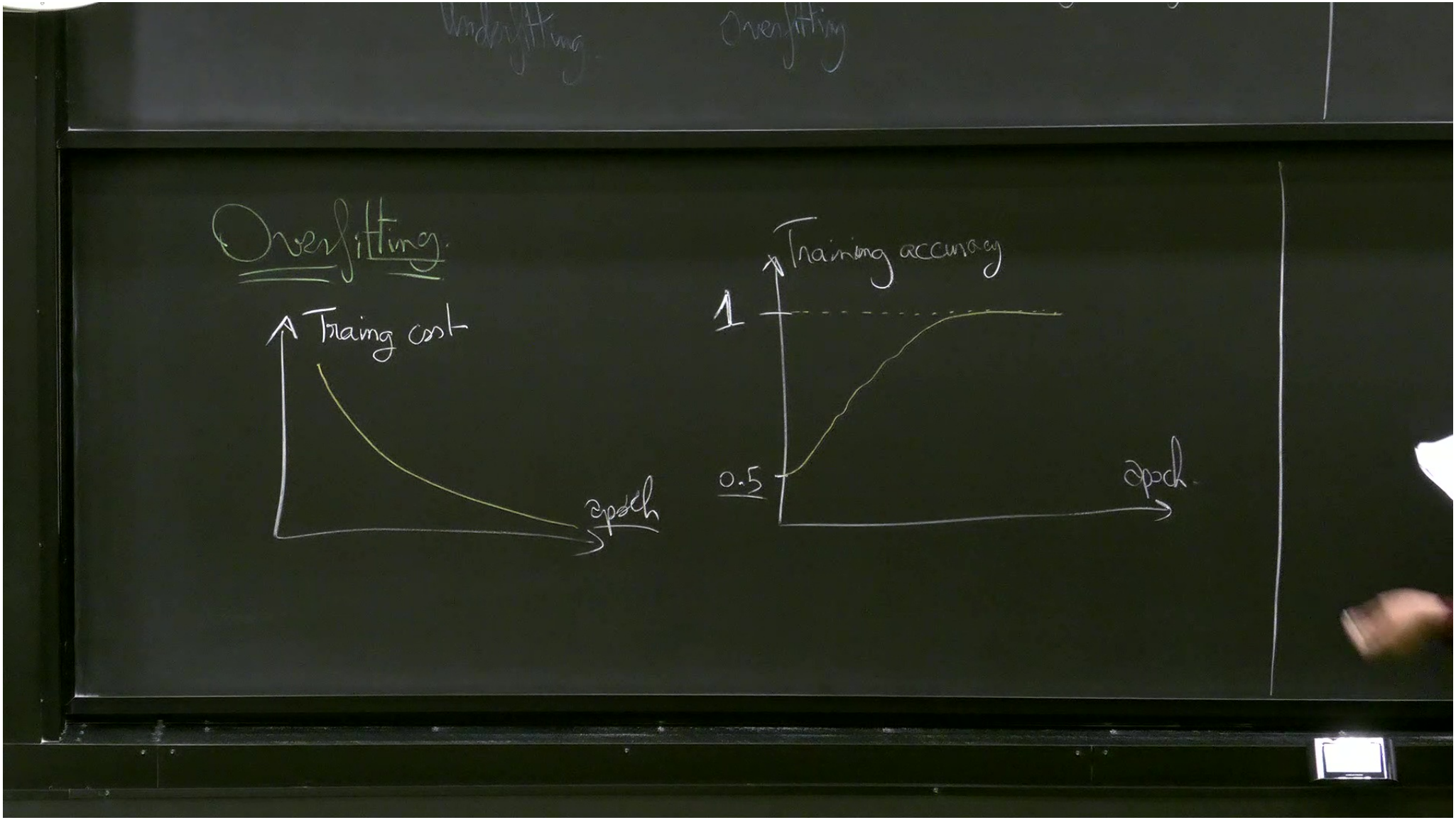
is equivalent to $\hat{\beta} = X^{\dagger}y$.

```
[ ]  1 import numpy as np
     2 import numpy.random as rand
     3 import numpy.linalg as la
     4 import pylab as plt
     5 import ipywidgets as widgets
     6 from ipywidgets import interact
```

```
[ ]  1 rseed = 1932
     2 rand.seed(rseed)
```

```
[ ]  1 n = 20
     2 sigma = 0.3 # stdev of noise
```

✓ 0s completed at 7:17 AM


Underfitting
Overfitting
Overfitting
Traing cost
epoch
Training accuracy
1
0.5
epoch

neurons).

→ The difference between E_{in} and E_{out} measures the difference between [illegible] "predicting".

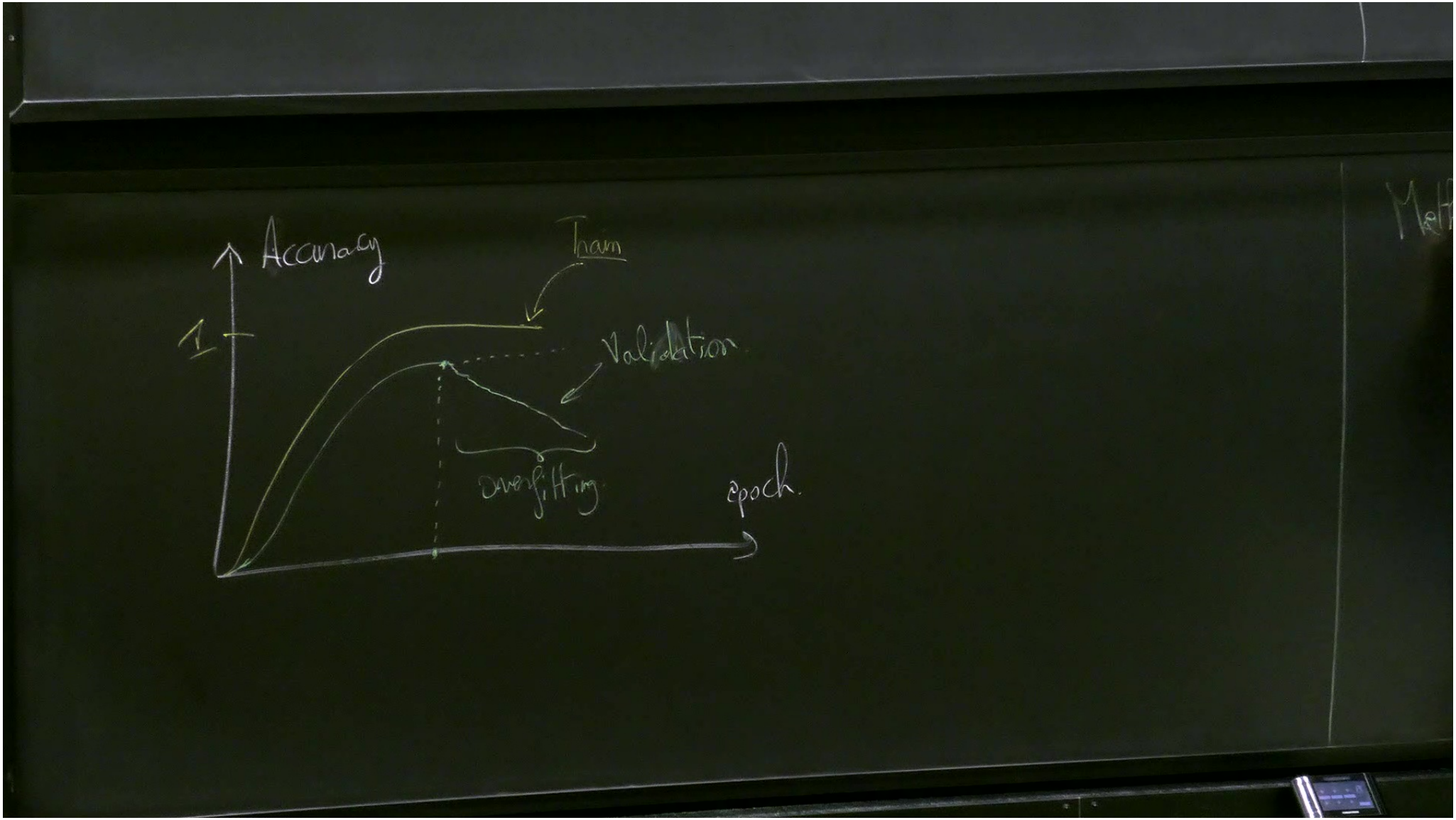
↳ If $E_{out} - E_{in} \gg 1$, the model is overfitting.

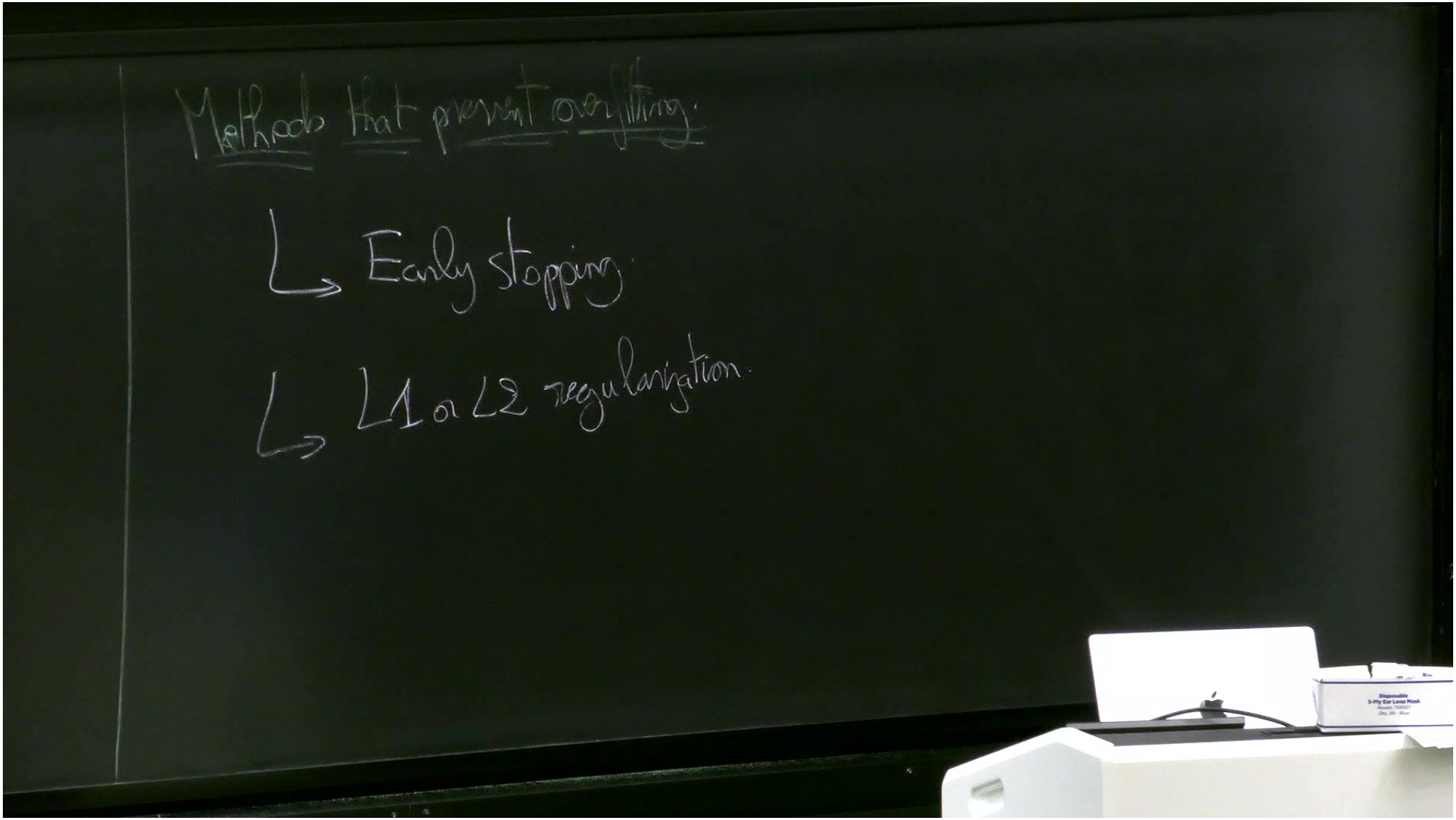
$$D = \{(\vec{x}, \vec{y})\}$$

Training dataset
D_{train} (70%)

Validation dataset
D_{val} (20%)
(Hyperparameter tuning)

Testing dataset
D_{test} (10%)
(Reporting the final Loss/accuracy)


Accuracy
Train
1
Validation
overfitting
epoch


Methods that prevent overfitting:
↳ Early stopping
↳ L1 or L2 regularization

Idea add an extra term to the cost function

$$C_{L2} = C_{init} + \lambda \sum_{\text{weights}} |w_{ij}^{(l)}|^2 \qquad (\lambda > 0)$$

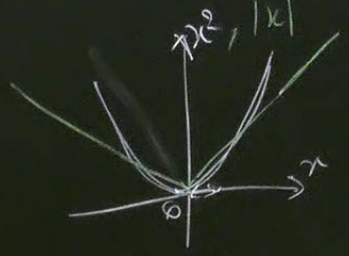
$$\sum_{\text{weights}} |w_{ij}^{(l)}|^2 = \|\text{weights}\|_2^2$$

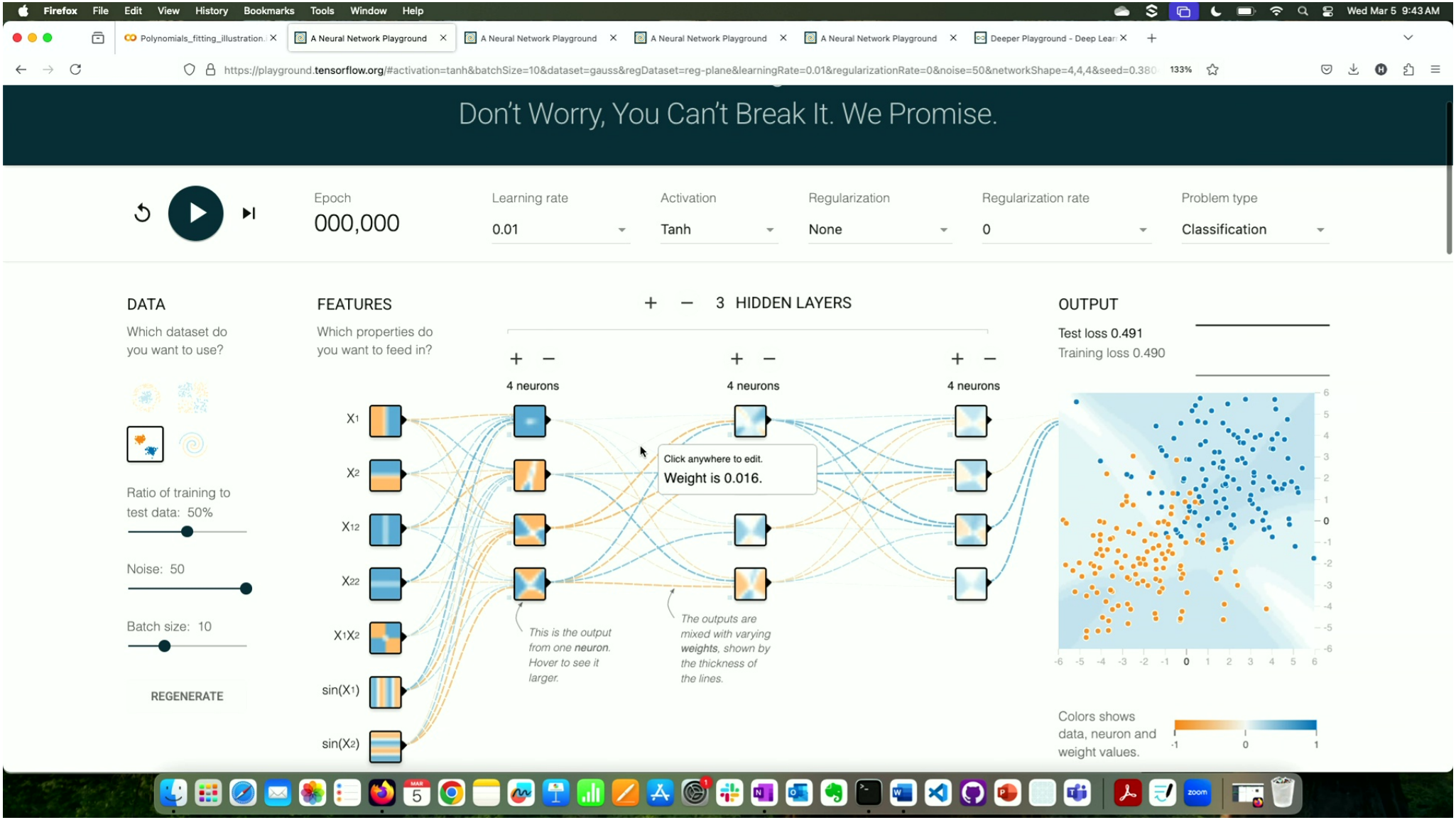
hyperparameter "L2 regularization"

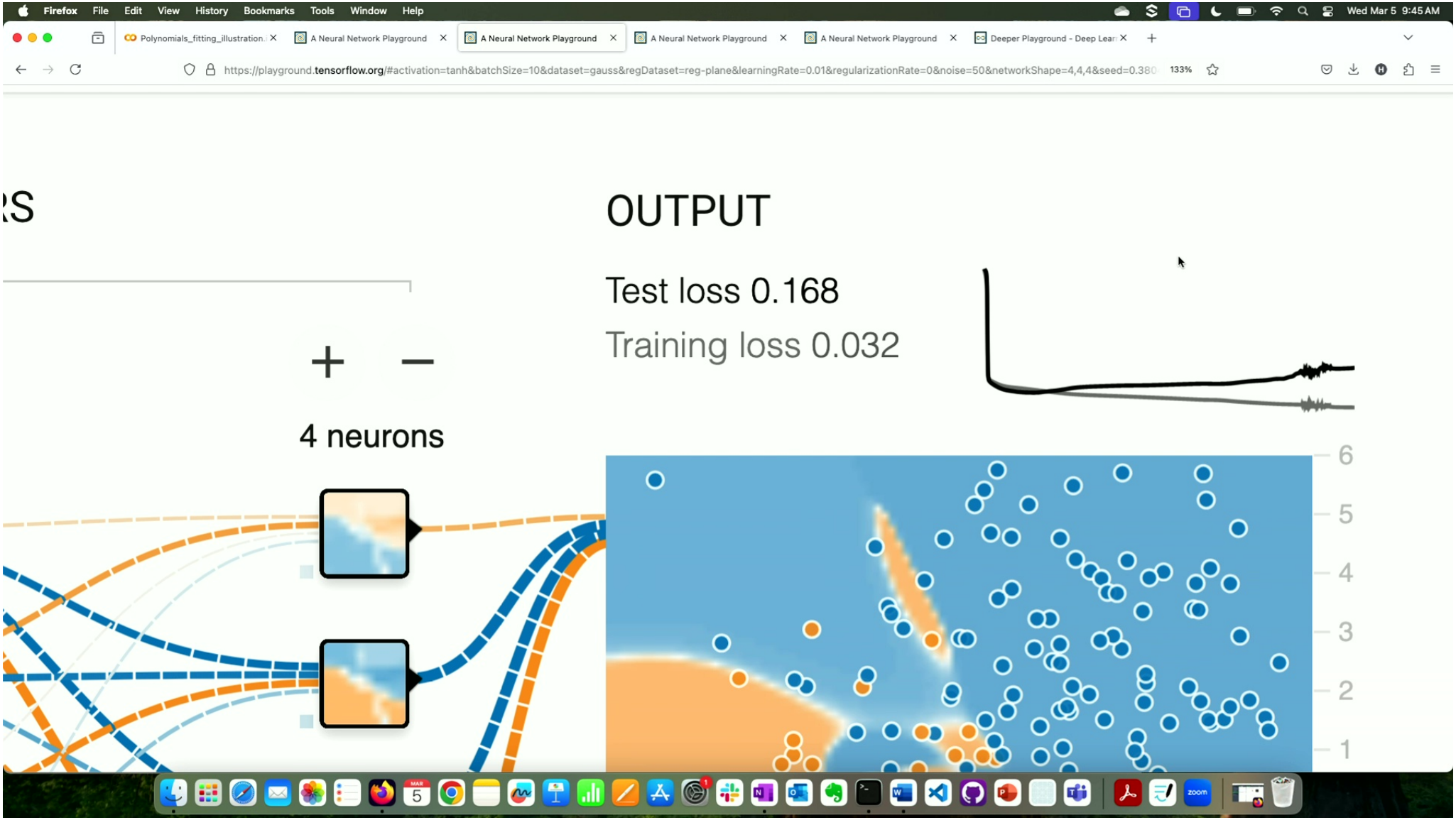
$$C_{L1} = C_{init} + \lambda \sum_{\text{weights}} |w_{ij}^{(l)}|$$

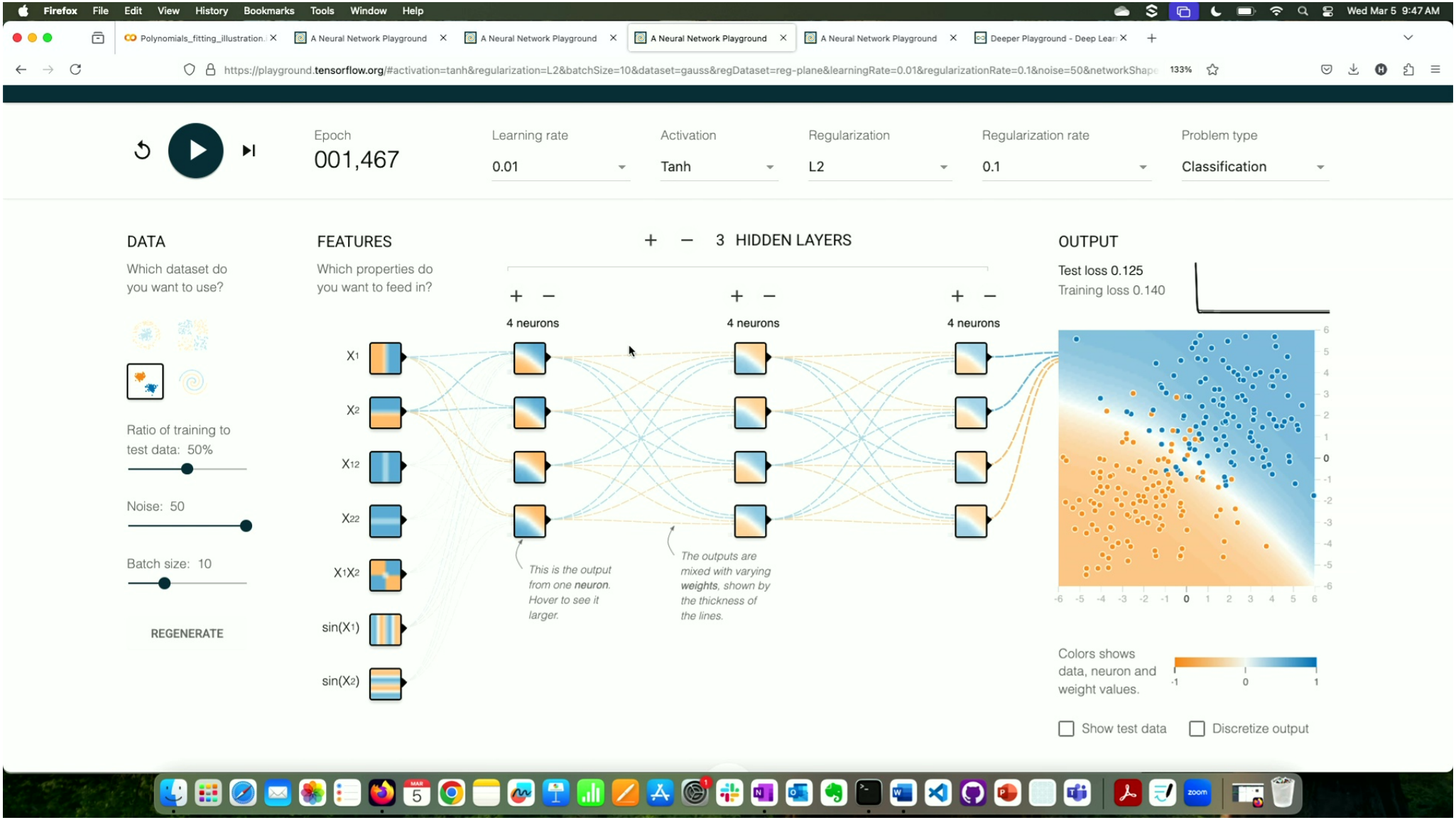
→ λ too small ⇝ overfitting

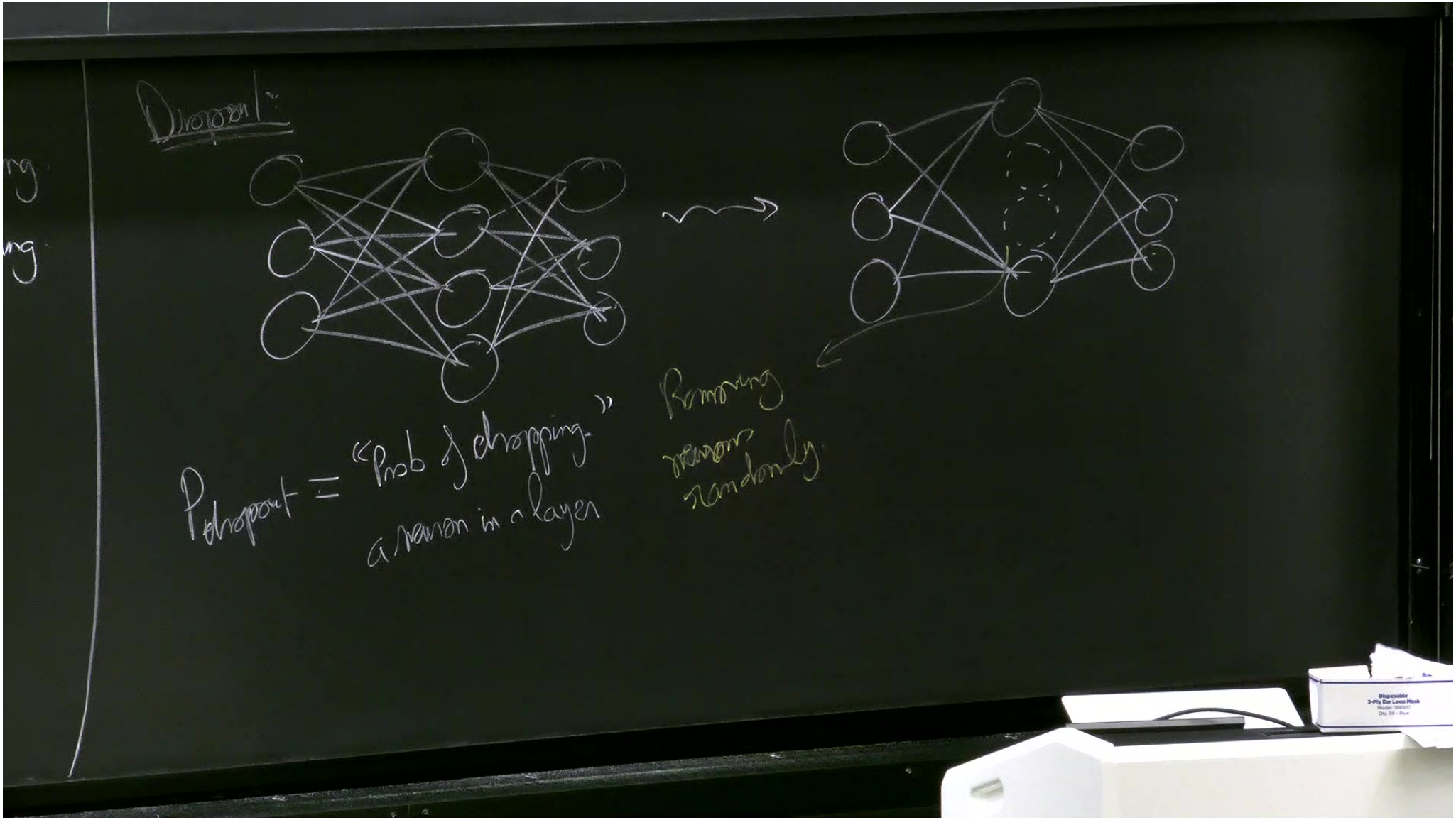
→ λ too small ⇝ underfitting

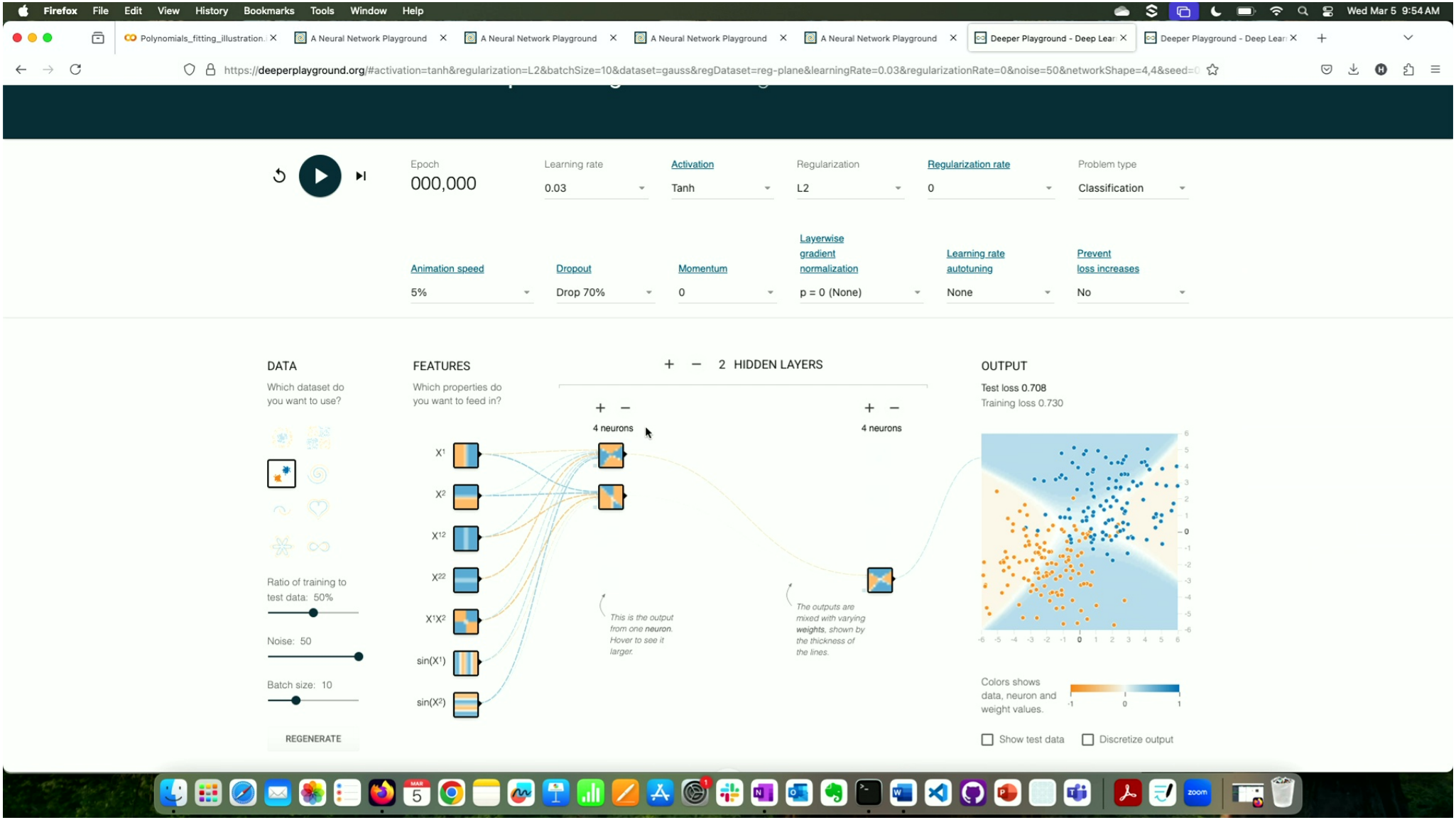


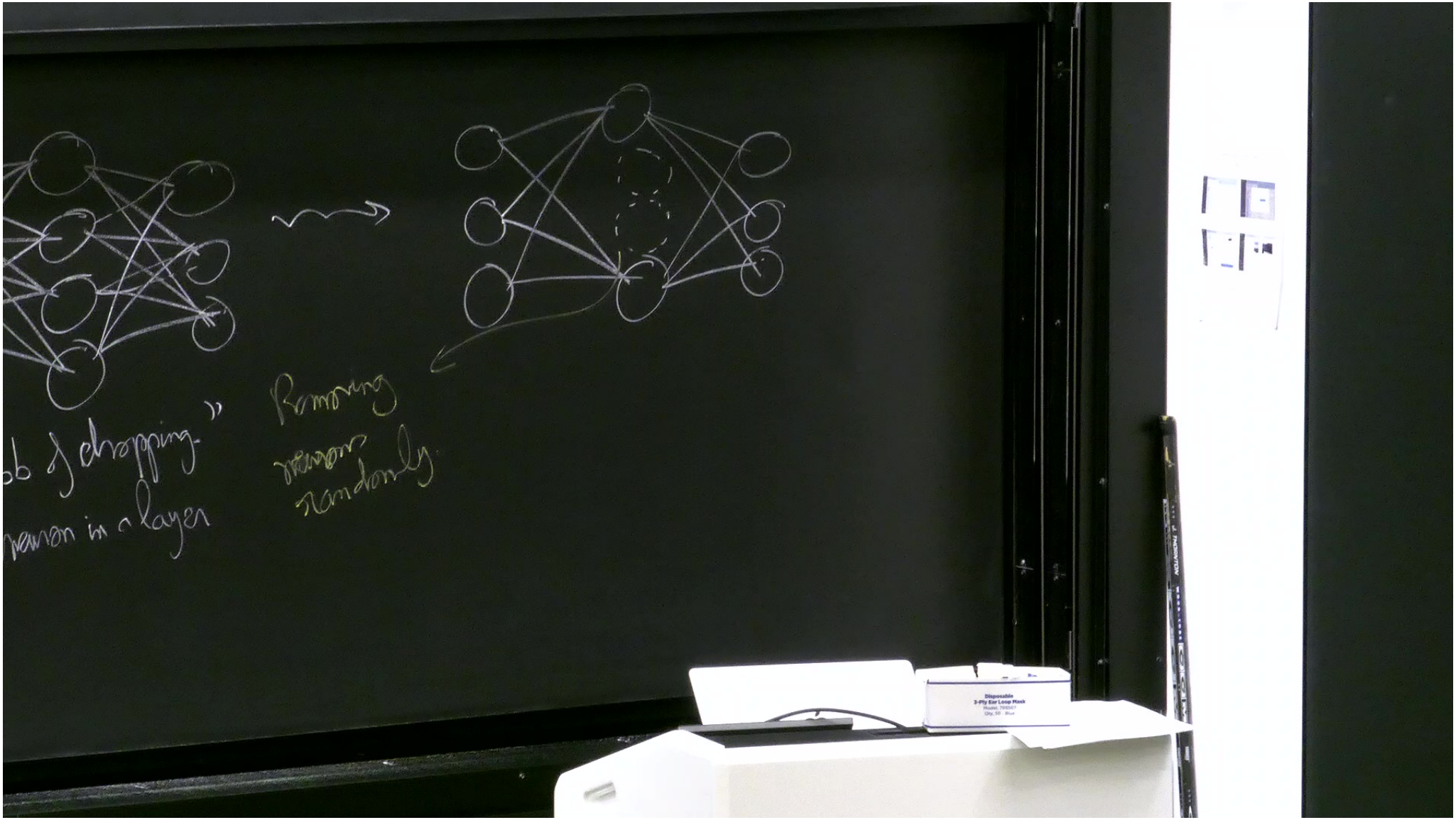

Don't Worry, You Can't Break It. We Promise.
Epoch
000,000
Learning rate
0.01
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 50
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X12
X22
X1X2
sin(X1)
sin(X2)
3 HIDDEN LAYERS
4 neurons
4 neurons
4 neurons
Click anywhere to edit.
Weight is 0.016.
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.491
Training loss 0.490
Colors shows data, neuron and weight values.


Firefox File Edit View History Bookmarks Tools Window Help
Wed Mar 5 9:45 AM
Polynomials_fitting_illustration.
A Neural Network Playground
A Neural Network Playground
A Neural Network Playground
A Neural Network Playground
Deeper Playground - Deep Lear
https://playground.tensorflow.org/#activation=tanh&batchSize=10&dataset=gauss&regDataset=reg-plane&learningRate=0.01&regularizationRate=0&noise=50&networkShape=4,4,4&seed=0.380
133%
OUTPUT
Test loss 0.168
Training loss 0.032
+
−
4 neurons
6
5
4
3
2
1


Epoch
001,467
Learning rate
0.01
Activation
Tanh
Regularization
L2
Regularization rate
0.1
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 50
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X12
X22
X1X2
sin(X1)
sin(X2)
3 HIDDEN LAYERS
4 neurons
4 neurons
4 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.125
Training loss 0.140
Colors shows data, neuron and weight values.
Show test data
Discretize output


Dropout
Prob of dropping
a neuron in a layer
Removing
neurons
randomly

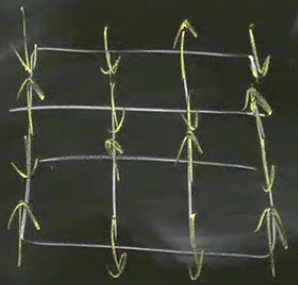

Firefox File Edit View History Bookmarks Tools Window Help
Wed Mar 5 9:54 AM
Polynomials_fitting_illustration.
A Neural Network Playground
A Neural Network Playground
A Neural Network Playground
A Neural Network Playground
Deeper Playground - Deep Lear
Deeper Playground - Deep Lear
https://deeperplayground.org/#activation=tanh&regularization=L2&batchSize=10&dataset=gauss&regDataset=reg-plane&learningRate=0.03&regularizationRate=0&noise=50&networkShape=4,4&seed=
Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
L2
Regularization rate
0
Problem type
Classification
Animation speed
5%
Dropout
Drop 70%
Momentum
0
Layerwise gradient normalization
p = 0 (None)
Learning rate autotuning
None
Prevent loss increases
No
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 50
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X12
X22
X1X2
sin(X1)
sin(X2)
2 HIDDEN LAYERS
4 neurons
4 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.708
Training loss 0.730
Colors shows data, neuron and weight values.
Show test data
Discretize output

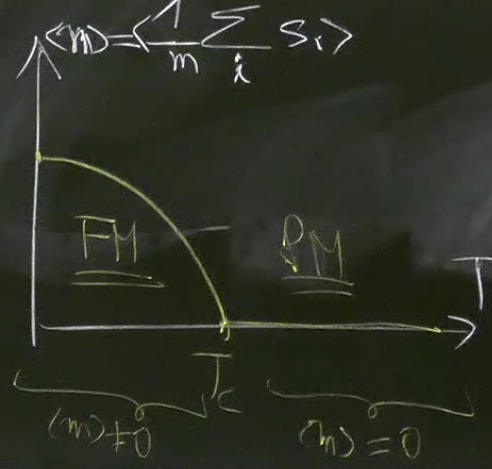

Removing
neurons
randomly

Underfitting Overfitting

Distinguishing classical phases of matter using SL:

$$H = -J \sum_{\langle ij \rangle} s_i s_j \qquad s_i = \pm 1$$





neurons).

→ The difference between E_{in} and E_{out} measures the [illegible] «predicting».

↳ If $E_{out} - E_{in} \gg 1$, the model is overfitting.

$$\vec{x} = (+1, -1, \cdots, +1)$$
$$s_1 \; s_2 \cdots s_N$$

Labels $y = 0$ or 1

0 ↓ FM 1 ↓ PM.