

Title: Machine Learning Lecture

Speakers: Mohamed Hibat Allah

Collection: Machine Learning 2023/24

Date: April 04, 2024 - 11:30 AM

URL: <https://pirsa.org/24040047>

Lecture 2

→ Acronyms so far

ML: Machine Learning

SL: Supervised Learning

UL: Unsupervised Learning

Outline for today

↳ Our first SL algorithm: linear regression

↳ Parameter optimisation: Gradient Descent

↳ Another SL algorithm: Logistic regression

Updates:

→ Regres

→ ML e

→ Recal

$\vec{x} = ($

↳ Datape

Updates:

→ Roger Malika guest lecture on Apr 25 (Lecture 11)

→ ML ethics + Your presentations (Lecture 13).

→ Recall from Lecture # 1, the setup of a SL problem $\rightarrow D = \{(\vec{x}, \vec{y})\}$ (Dataset).

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

↓
Data points

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

↓
Labels

Task: Fit some function $f(\vec{x})$ to \vec{y}
with $f: \mathbb{R}^{d_x} \rightarrow \mathbb{R}^{d_y}$

$f(\vec{x}) \approx \vec{y}$

* Linear regression (LR)

$$D = \{(\vec{x}, y)\} \rightarrow \text{Fit } f(\vec{x}) \rightarrow y \text{ where}$$
$$f(\vec{x}) = \sum_{j=1}^{d_x} \omega_j x_j = \omega^T \vec{x}$$

$$d_y = 1$$

If $d_y > 1 \rightarrow$ Multivariate LR

$M = \#$ of datapoints in D .

$x_j^{(i)}$ = j^{th} element of $\vec{x}^{(i)}$

$1 \leq j \leq d_x$

$X \in \mathbb{R}^{M \times d_x}$ $(X)_{ij} = x_j^{(i)}$

Label vector $\vec{y} = (y^{(1)}, y^{(2)}, \dots, y^{(M)})$

Loss function, \mathcal{L}

$$\mathcal{L} = \sum_{i=1}^M \left(\sum_{j=1}^d w_j x_j^{(i)} - y^{(i)} \right)^2$$

$$= \| X \vec{w} - \vec{y} \|_2^2$$

$\vec{w} \in \mathbb{R}^{d_x}$ = argmin \mathcal{L}

Label vector $\vec{y} = (y^1, y^2, \dots, y^m)$

$\vec{w} \in \mathbb{R}^n$

$$\begin{aligned} \mathcal{L} &= (\mathbf{X}\vec{w} - \vec{y})^T (\mathbf{X}\vec{w} - \vec{y}) \\ &= \vec{w}^T \mathbf{X}^T \mathbf{X} \vec{w} - \vec{y}^T \mathbf{X} \vec{w} - \vec{w}^T \mathbf{X}^T \vec{y} + \vec{y}^T \vec{y} \end{aligned}$$

$$0 = \frac{\partial \mathcal{L}}{\partial w_i} = 2(\mathbf{X}^T \mathbf{X} \vec{w})_i - 2(\mathbf{X}^T \vec{y})_i \quad \forall i$$

$$\boxed{(\mathbf{X}^T \mathbf{X}) \vec{w}_{\mathbb{R}} = \mathbf{X}^T \vec{y}}$$

$\vec{w}_{\mathbb{R}} =$

$$w_{LR} = (X^T X)^{-1} (X^T Y)$$

$$f_{DLR}(\vec{x}) = w_{LR}^T \vec{x} \approx y$$

If $X^T X$ is not invertible \rightarrow ∞ many solutions.
 $\text{Rank}(X) < dx$

Regression

→ Gradient Descent (GD)

↳ Can be more efficient when d_n is large.

↳ Broadly useful in ML.

←
Data points

↓
Labels

with $f: \mathbb{R} \rightarrow \mathbb{R}$

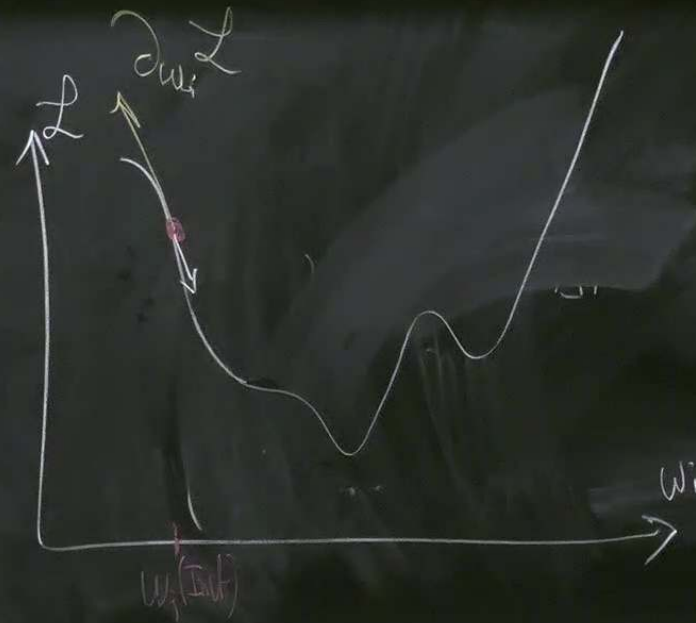
Given L (loss function)

$$\vec{w} = (w_1, w_2, \dots, w_n); \quad \frac{\partial L}{\partial w_i} = 0$$

For LR $n = dx$

For other ML apps $n \neq dx$ *learning rate*

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



$$\mathcal{L}(w_i + \Delta w_i) = \mathcal{L}(w_i) + \Delta w_i \partial_{w_i} \mathcal{L} + O((\Delta w_i)^2)$$

$$\mathcal{L}(w_i + \Delta w_i) = \mathcal{L}(w_i) - \underbrace{\eta (\partial_{w_i} \mathcal{L})^2}_{\leq 0} + \underbrace{O((\Delta w_i)^2)}_{O(\eta^2)}$$

Hyperparameter

• η small enough

• η not too small not too big

$$\mathcal{L}(w_i + \Delta w_i) \leq \mathcal{L}(w_i)$$

$$\Delta w_i = -\eta \times \partial_{w_i} \mathcal{L}$$



Logistic regression (Binary classification)

$$y = 0, 1$$

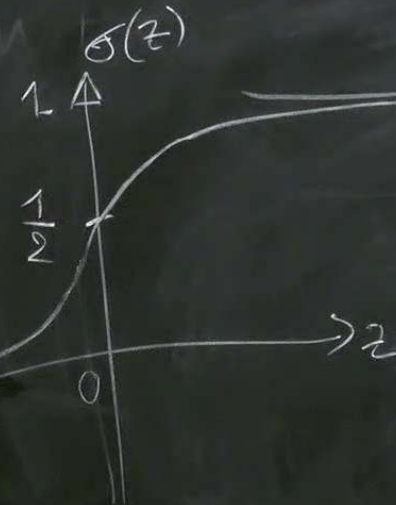
Given some dataset $D = \{(\vec{x}, y)\}$

$$z = b + w_1 x_1 + w_2 x_2 + \dots + w_n x_n$$

Bias

Weights

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

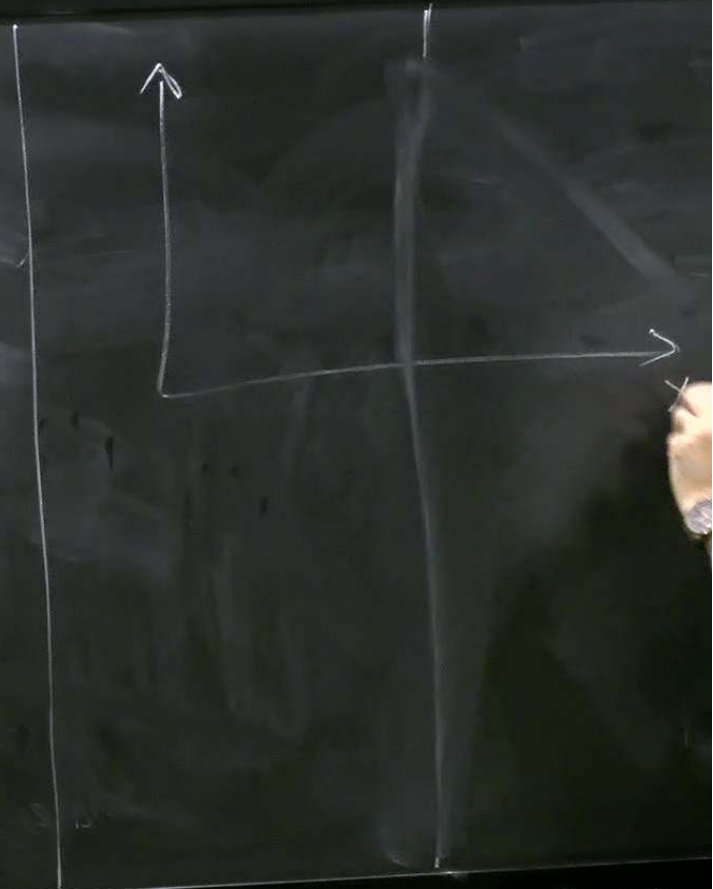
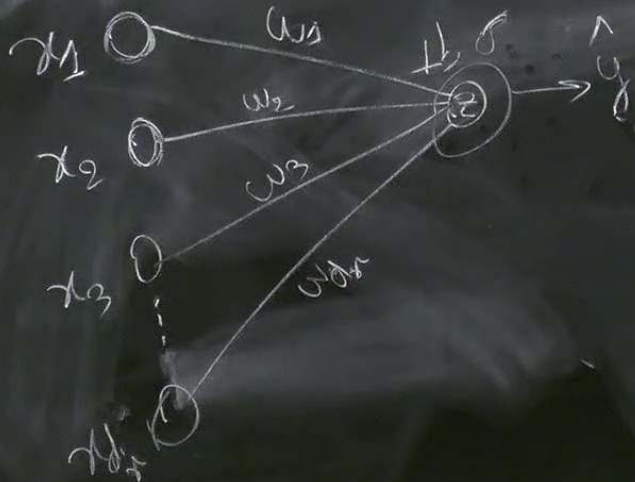


sigmoid
function

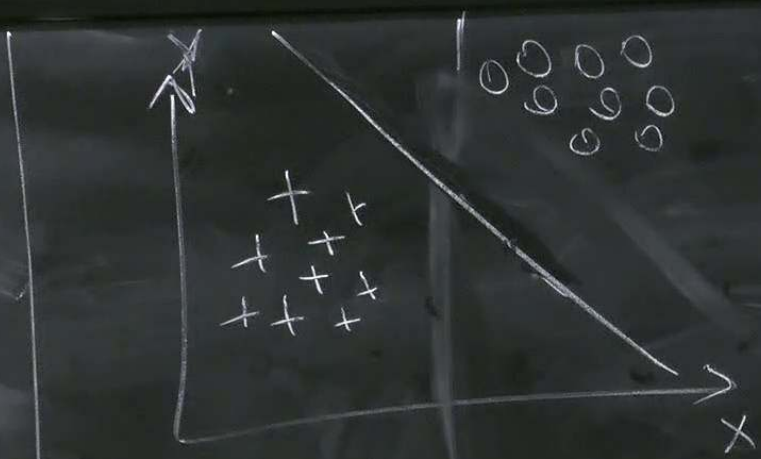
label vector $\vec{y} = (y^{(1)}, y^{(2)}, \dots, y^{(n)})$

$w \in \mathbb{R}^{d \times 1}$

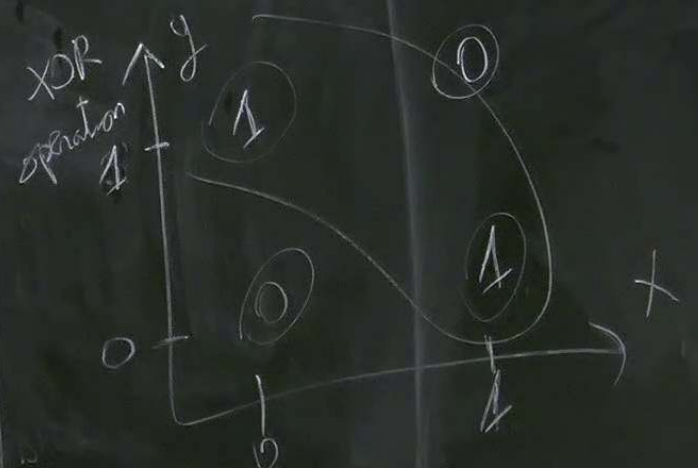
$\hat{y} \geq 0.5 \rightarrow \text{prediction} = 1$
 $\hat{y} < 0.5 \rightarrow \text{prediction} = 0$

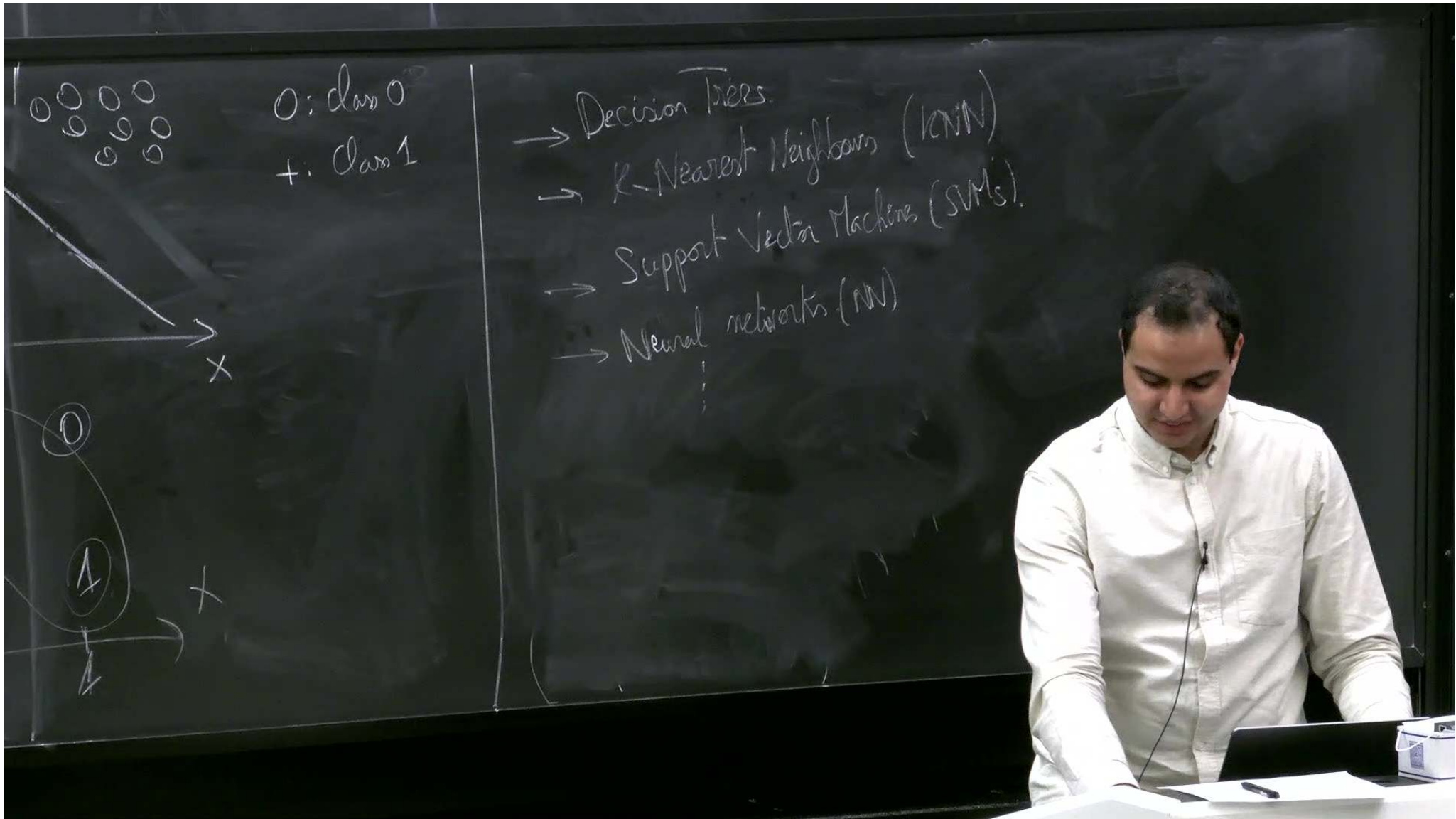


$w_{LP} = \text{argmin}$
 $w \in \mathbb{R}^{d_x}$



o: class 0
+: class 1





0: class 0
+ class 1

- Decision Trees
- K-Nearest Neighbors (KNN)
- Support Vector Machines (SVMs)
- Neural networks (NN)

Firefox File Edit View History Bookmarks Tools Window Help

Machine Learning for Many-Bo... A Neural Network Playground

https://playground.tensorflow.org/#activation=sigmoid&batchSize=15&dataset=gauss®Dataset=reg-plane&learningRate=0.03®ularizationRate=0&noise=0&networkShape=&se...

110% ☆

Thinker with a Neural Network Right Here in Your Browser.

Don't Worry, You Can't Break It. We Promise.

↻ ▶ 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

REGENERATE

FEATURES
Which properties do you want to feed in?

+ - 0 HIDDEN LAYERS

X¹ X² X¹² X²² X¹X² sin(X¹) sin(X²)

OUTPUT
Test loss 0.584
Training loss 0.580

Colors shows data, neuron and weight values.

Show test data Discretize output

Firefox File Edit View History Bookmarks Tools Window Help

PSI Portal Machine Learning for Many-Bo A Neural Network Playground A Neural Network Playground

https://playground.tensorflow.org/#activation=sigmoid&batchSize=15&dataset=xor®Dataset=reg-plane&learningRate=0.03®ularizationRate=0&noise=0&networkShape=&seed= 110%

Epoch: 001,302 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 REGENERATE

FEATURES: Which properties do you want to feed in? X1, X2, X1², X2², X1X2, sin(X1), sin(X2)

0 HIDDEN LAYERS

OUTPUT: Test loss 0.512 Training loss 0.507

Colors shows data, neuron and weight values. -1 0 1

Show test data Discretize output

Thinker with a **Neural Network** Right Here in Your Browser.
 Don't Worry, You Can't Break It. We Promise.

Epoch: **000,512** 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

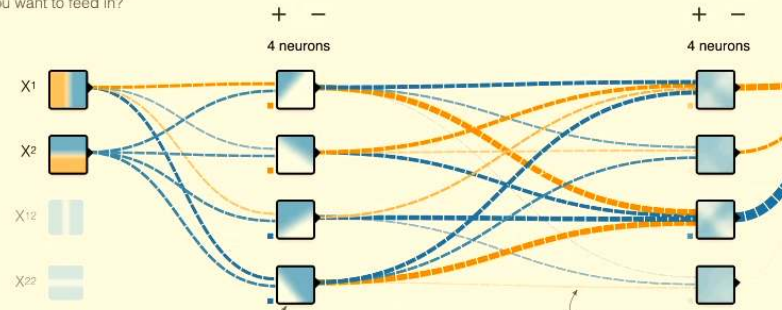
Batch size: 15

REGENERATE

FEATURES
 Which properties do you want to feed in?

- X¹
- X²
- X¹²
- X²²
- X¹X²
- sin(X¹)
- sin(X²)

+ - 2 HIDDEN LAYERS

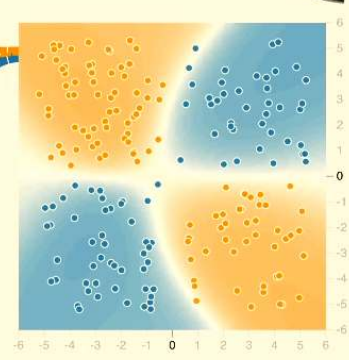


This is the output from one neuron. Hover to see it larger.

The outputs are mixed with varying weights, shown by the thickness of the lines.

OUTPUT

Test loss 0.069
 Training loss 0.053



Colors shows data, neuron and weight values.

Show test data Discretize output



Thinker with a **Neural Network** Right Here in Your Browser.
 Don't Worry, You Can't Break It. We Promise.

Epoch 002,165 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

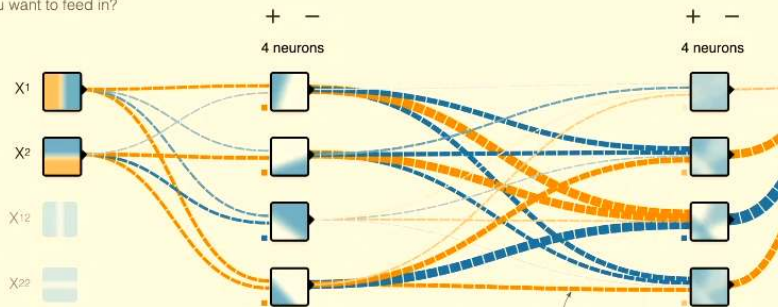
REGENERATE

FEATURES

Which properties do you want to feed in?

- X¹
- X²
- X¹²
- X²²
- X¹X²
- sin(X¹)
- sin(X²)

2 HIDDEN LAYERS

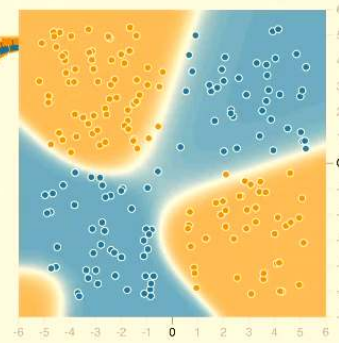


This is the output from one neuron. Hover to see it larger.

The outputs are mixed with varying weights, shown by the thickness of the lines.

OUTPUT

Test loss 0.063
 Training loss 0.020



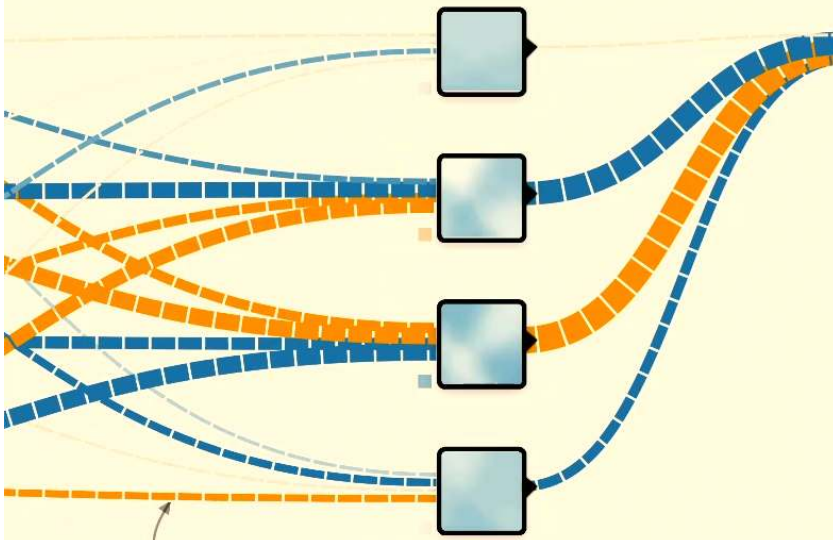
Colors shows data, neuron and weight values.

Show test data Discretize output

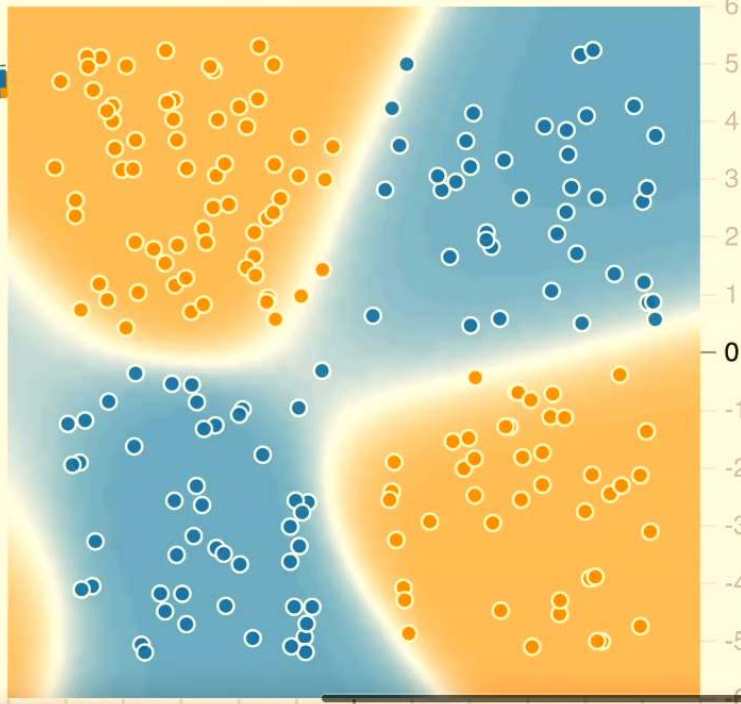
HIDDEN LAYERS

OUTPUT

+ -
4 neurons



Test loss 0.057
Training loss 0.029



The outputs are mixed with varying weights, shown by