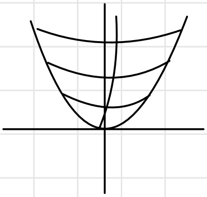
3/28 Neural Network

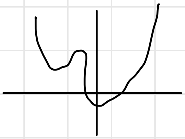
Gradient Descent



typically can find a local min / max, not always find global min / max

Stochastic GD

Goal: approximate of the cost function



locally maximize the cost

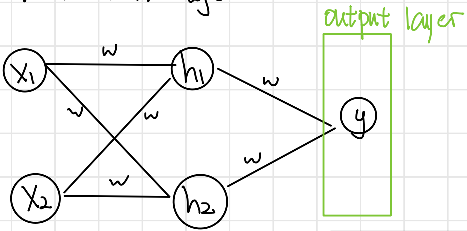
Intro to Neural Networks

Logistic Regression · fell short for XOR. couldn't find the line but works for OR. it is linearly separable

- $\text{XOR}(x_1, x_2) = h_1 \vee h_2$

Forward Propagation

3 models to work tgt



using x_1, x_2

Use logistic Regression to predict h_1, same for h_2

Then, use h_1, h_2 to predict y

Feedforward NN

in each layer, learning a speafic feature.

matrix notation:
$$\begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \sigma\left(\begin{bmatrix} w_{00} & w_{01} \\ w_{10} & w_{11} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}\right)$$

$$y = \sigma\left(\begin{bmatrix} w_{00} \\ w_{01} \end{bmatrix}^T \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} + b^{(2)}\right)$$

BackPropagation NN

from RHS → LHS

$$\text{Cost}(w, b) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log(\sigma(-w^T x_i + b)) + (1-y_i)\log(1-\sigma(-w^T x_i + b))\right]$$

log likelihood ⇒

Tunning Parameters

8 steps.