



Machine Learning Course

Workshop II: Neural Networks

Slides: Christof Peter, Tibor Cornelli

In Cooperation with

Prof. Dr. Martin Schlather
Chair in Applied Stochastics

Prof. Dr. Leif Döring
Chair in Stochastics



Course structure

Part I: Basics of
Machine
Learning

04.10 – Introduction to machine learning
11.10 – Neural networks

Part II:
Computer Vision
(CV)

18.10 – Convolutional Neural Networks
25.10 – Practical application of CNNs
01.11 – CV Challenge presentation

Part III: Natural
Language
Processing
(NLP)

08.11 – Recurrent Neural Networks
15.11 – Practical application of RNNs
22.11 – NLP Challenge Presentation

Agenda



STADS



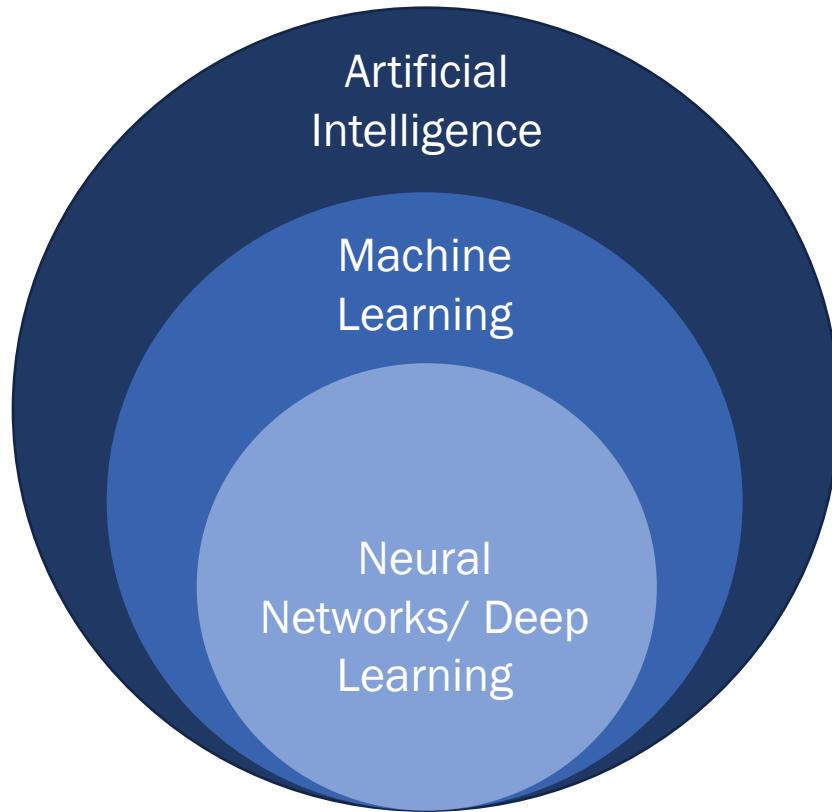
Intro

How do neural networks work?

Practical implementation

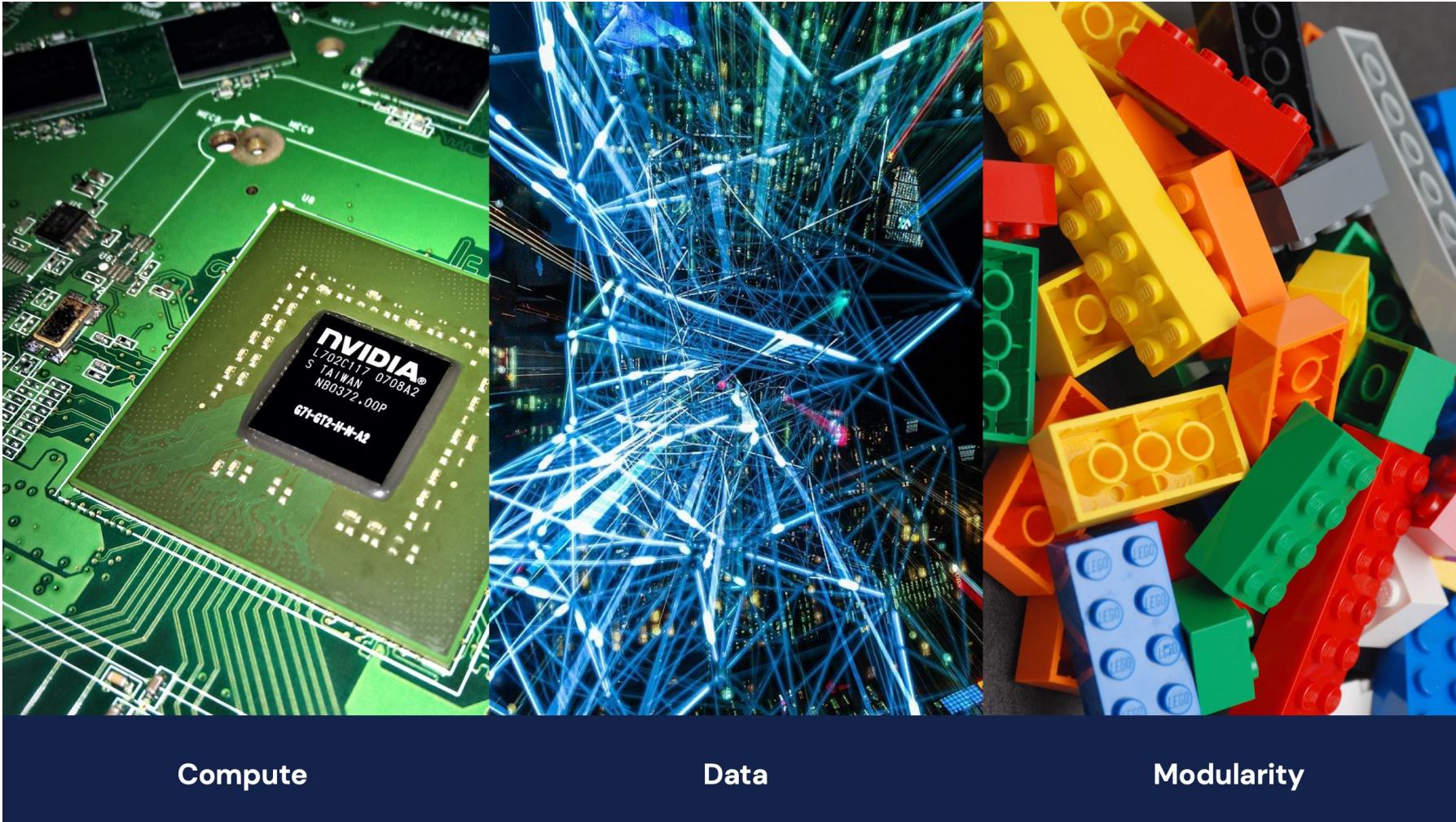
Classification & terminology

Our focus today lies on ANN's



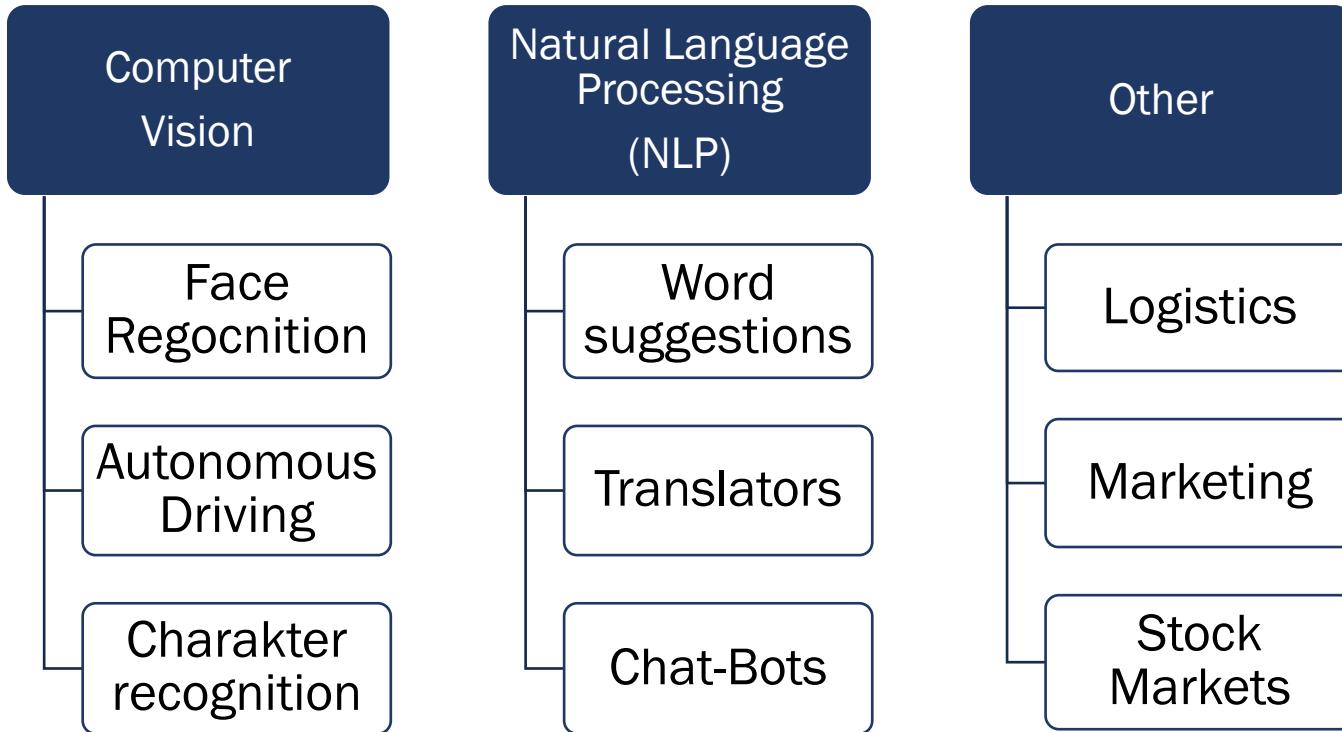
AI	Artificial Intelligence
ML	Machine Learning
ANN	Artificial Neural Network
DNN	Deep Neural Network
CNN	Convolutional Neural Network

History

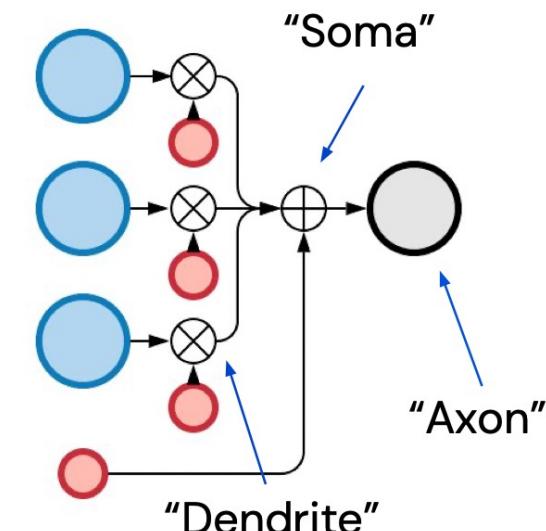
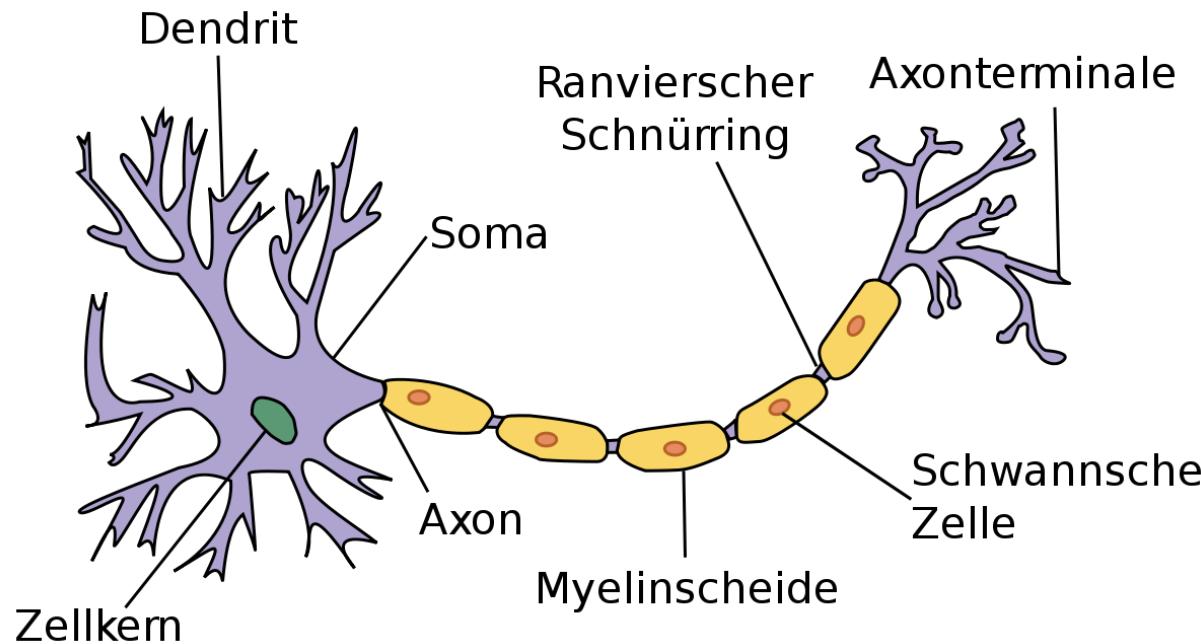


What are NNs used for?

Artificial neural networks have many applications. Currently, computer vision and natural language processing are the hottest topics.



Why “neural”?

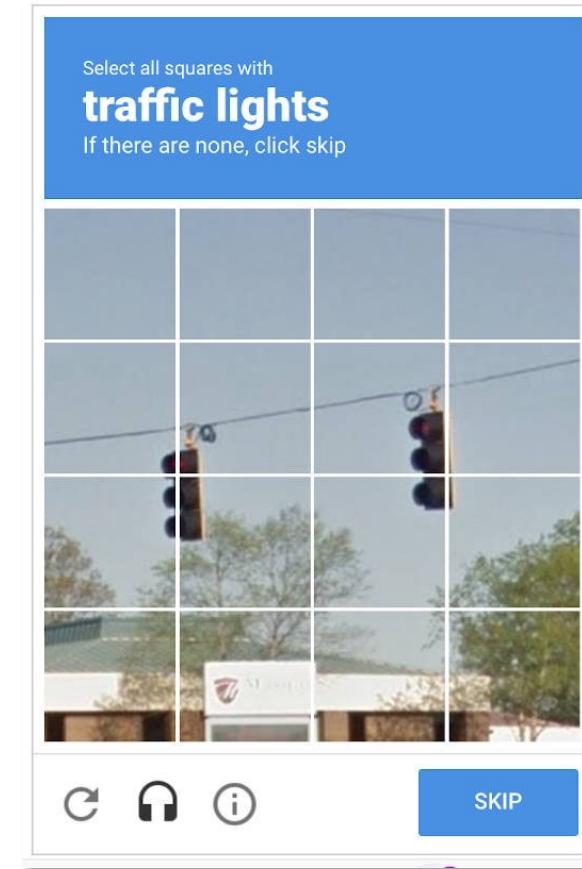
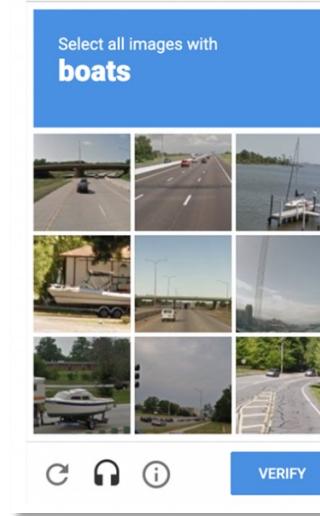
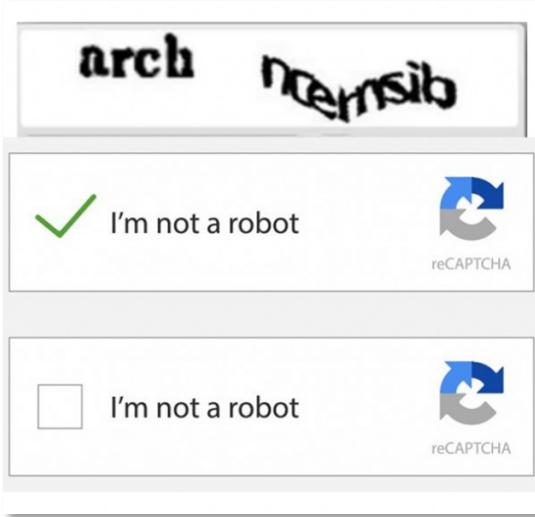


$$\sum_{i=1}^d \mathbf{w}_i \mathbf{x}_i + b$$

$$\sum_{i=0}^d \mathbf{w}_i \mathbf{x}_i \quad \mathbf{x}_0 := 1$$

Excursion: Recaptcha

Quelle: <https://www.techradar.com/news/captcha-if-you-can-how-youve-been-training-ai-for-years-without-realising-it>



Agenda



STADS

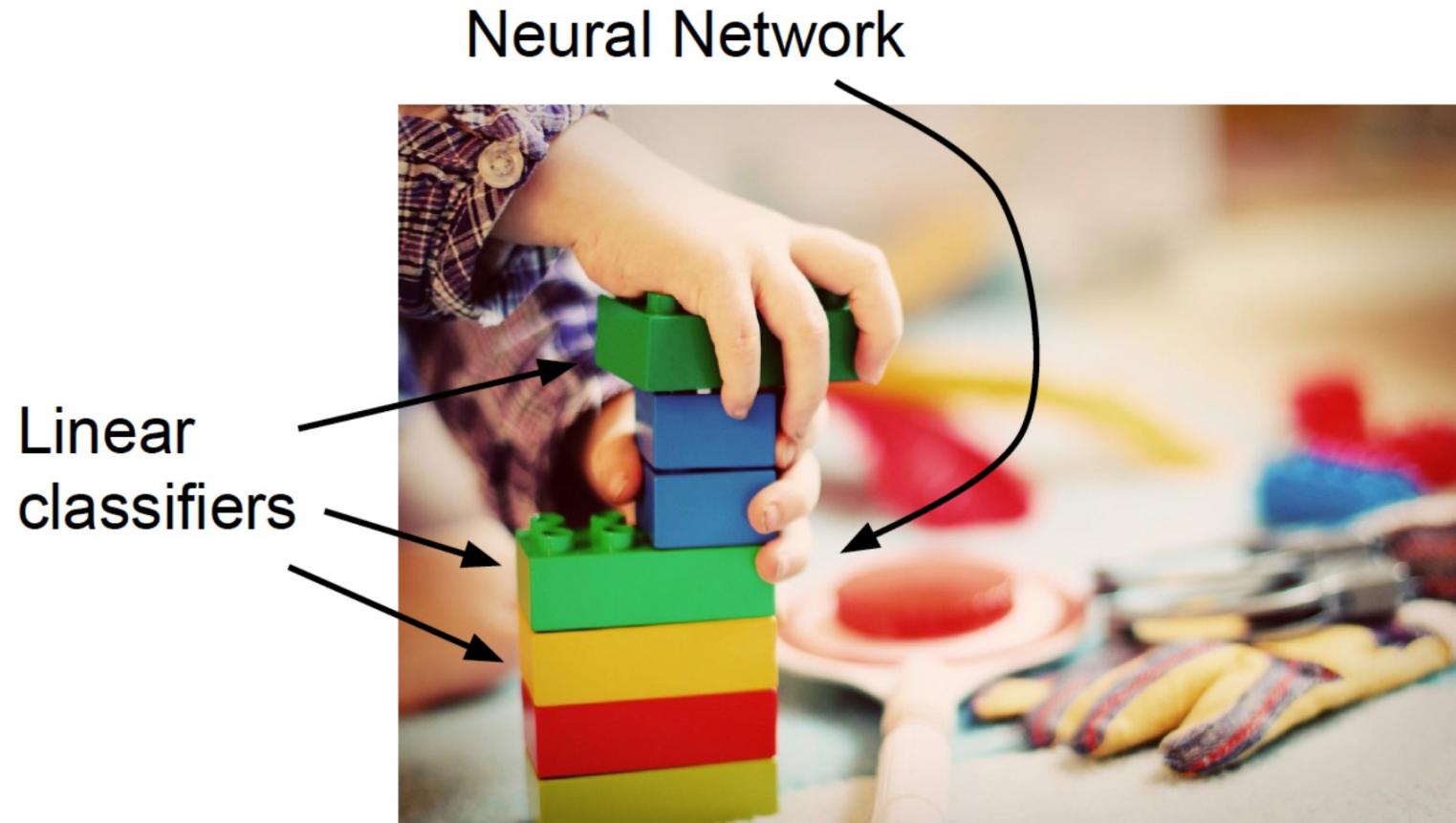
Intro



How do neural Networks work?

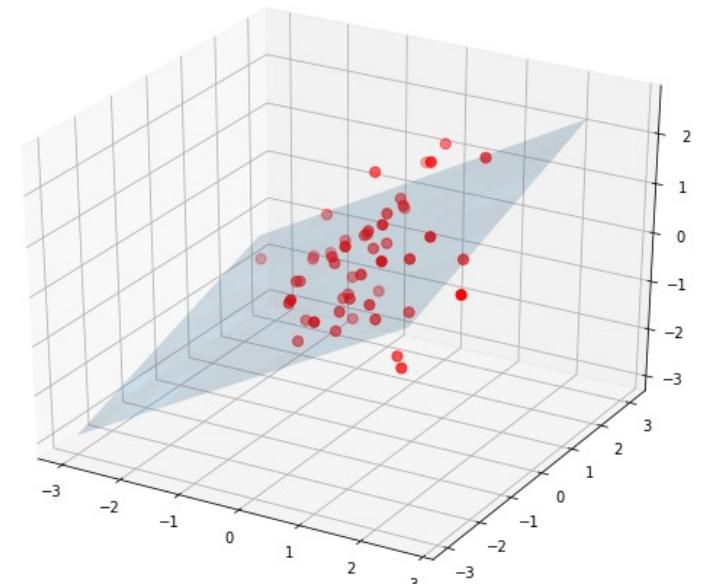
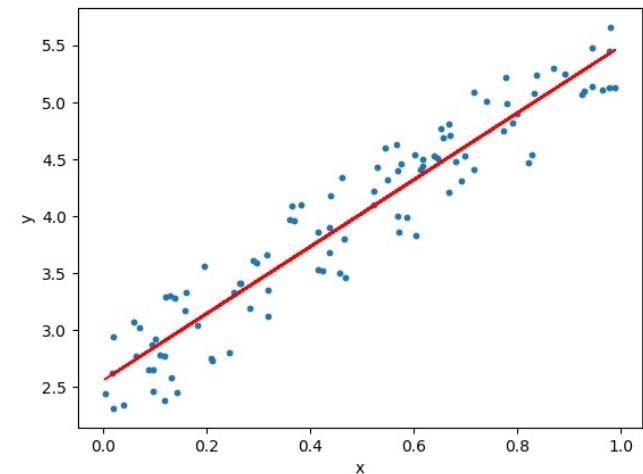
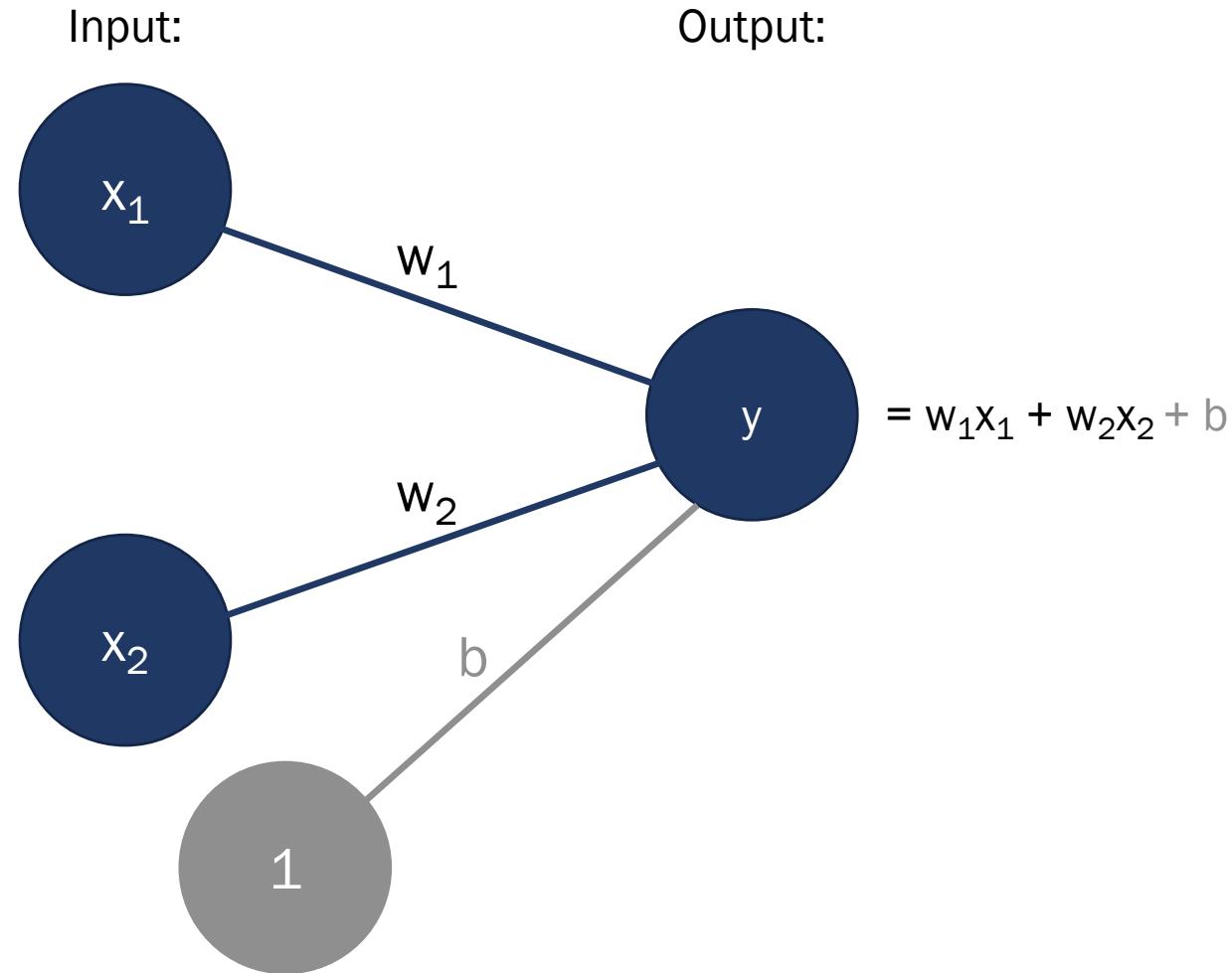
Practical implementation

Idea behind Neural Networks



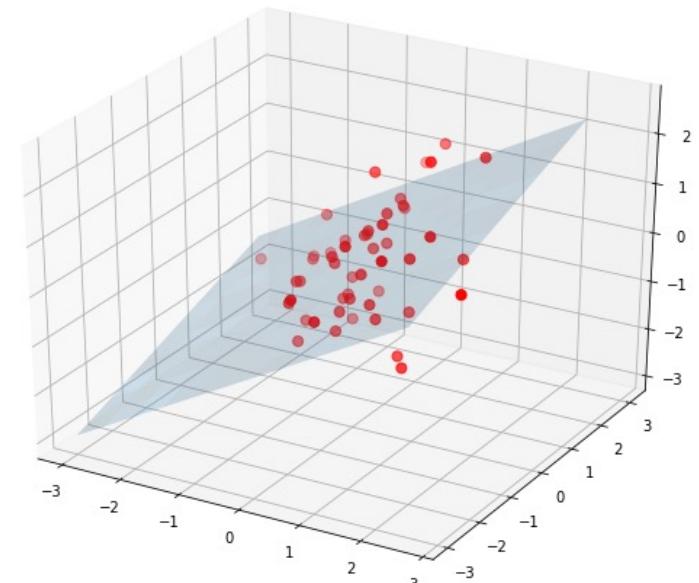
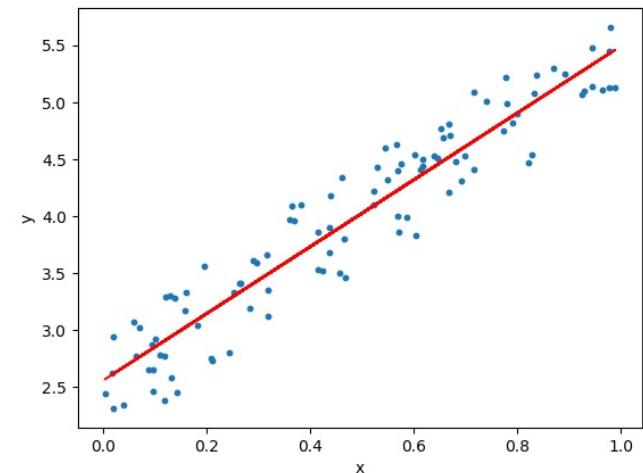
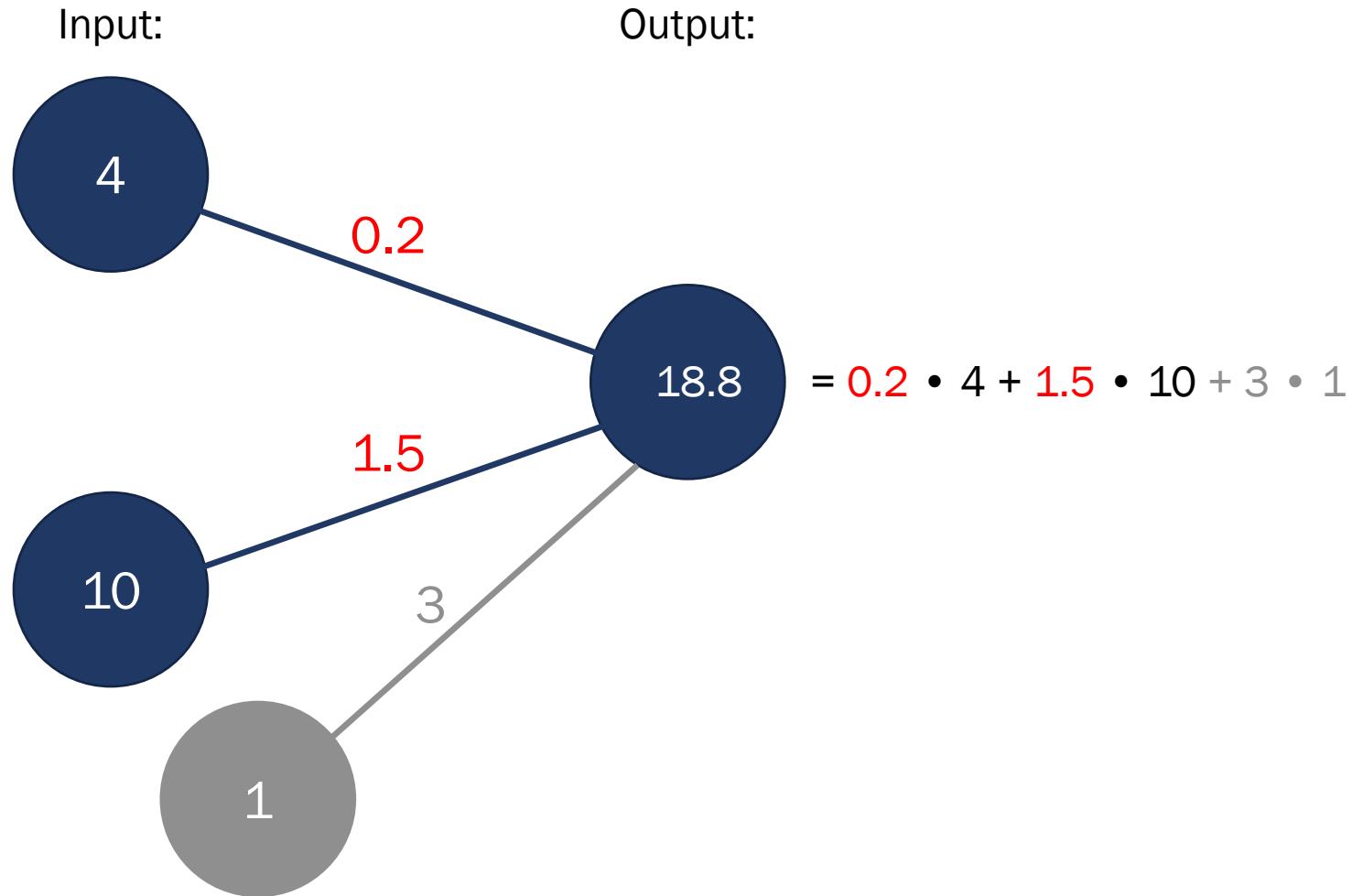
Linear Regression

Predictions general case

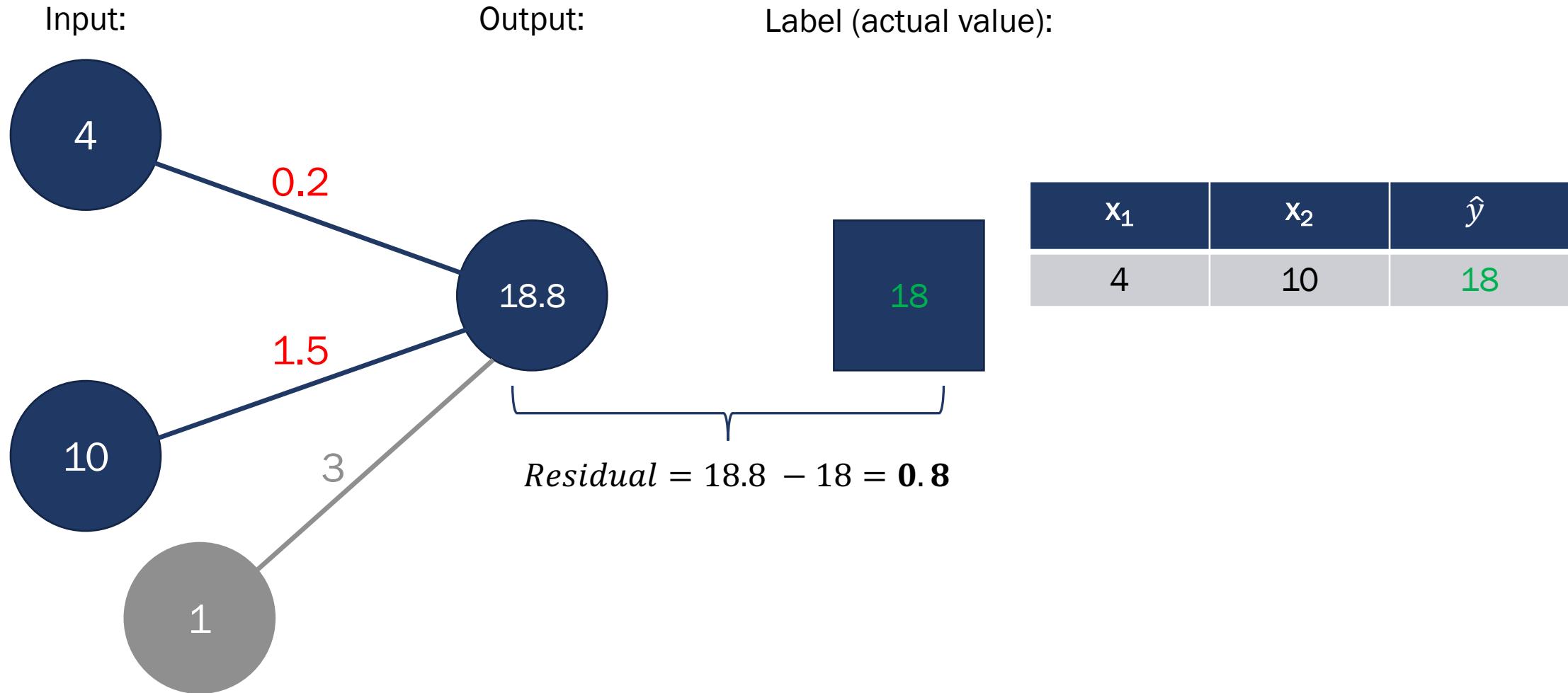


Linear Regression

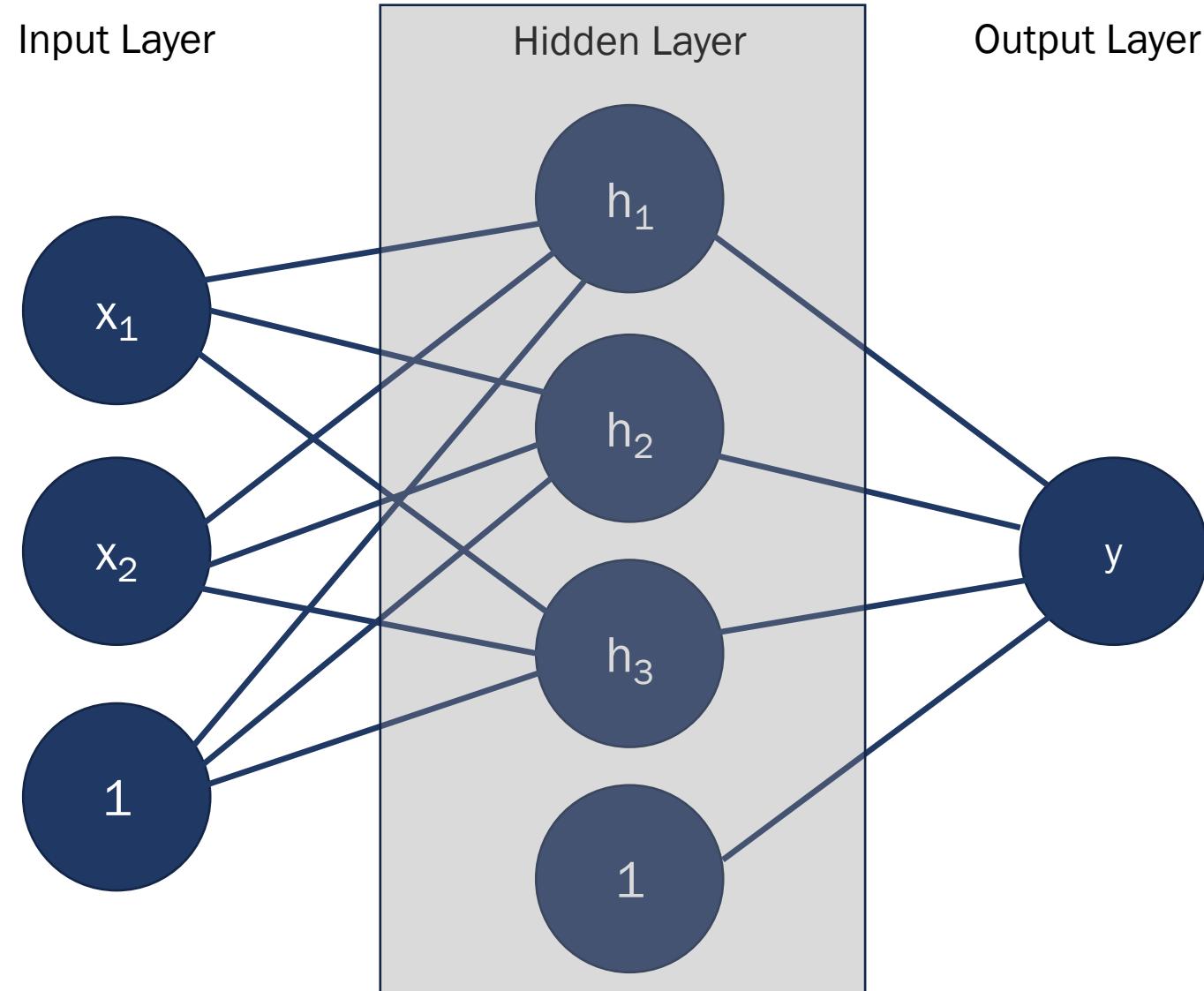
Predictions example



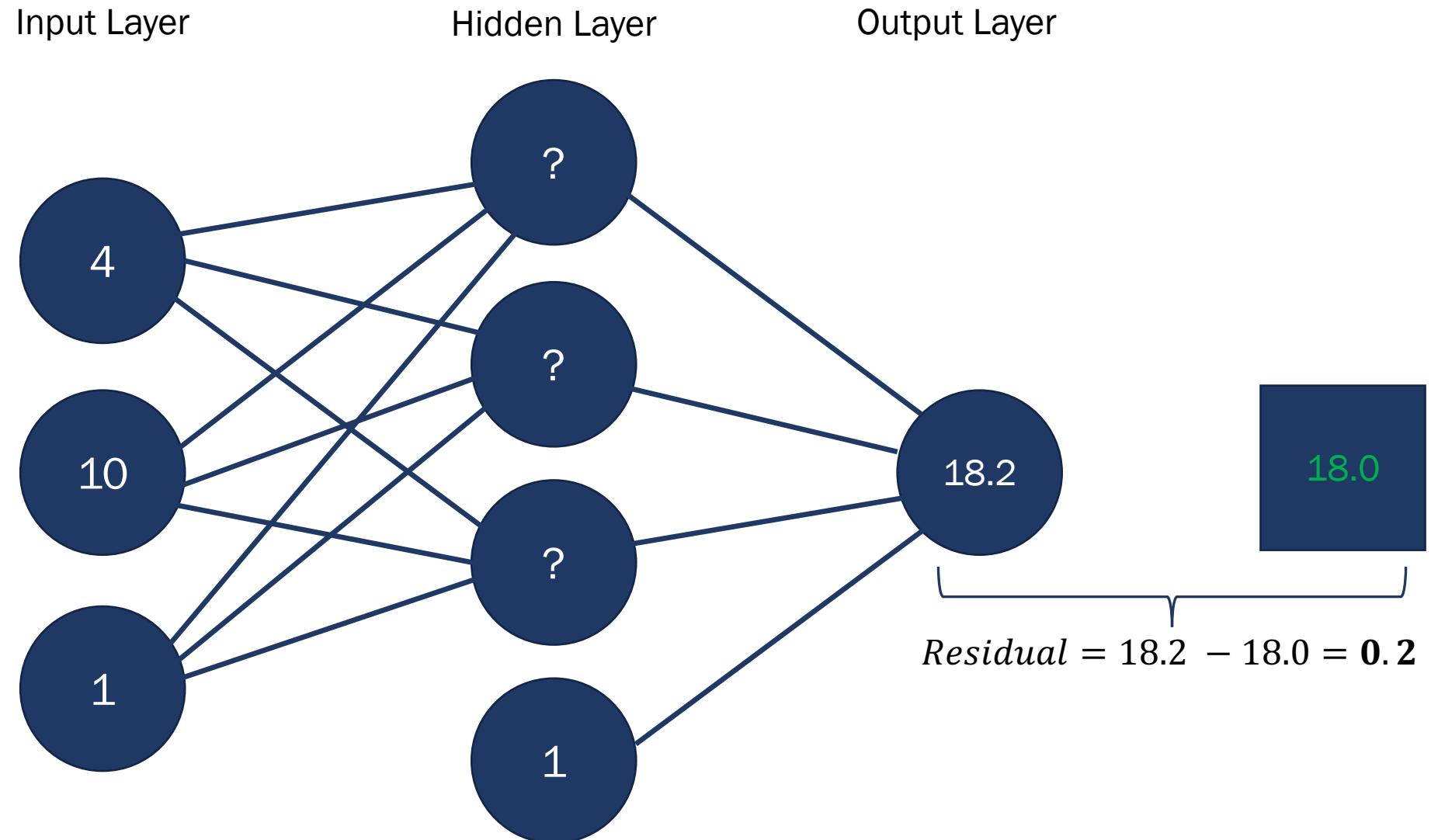
Linear Regression: Outcome



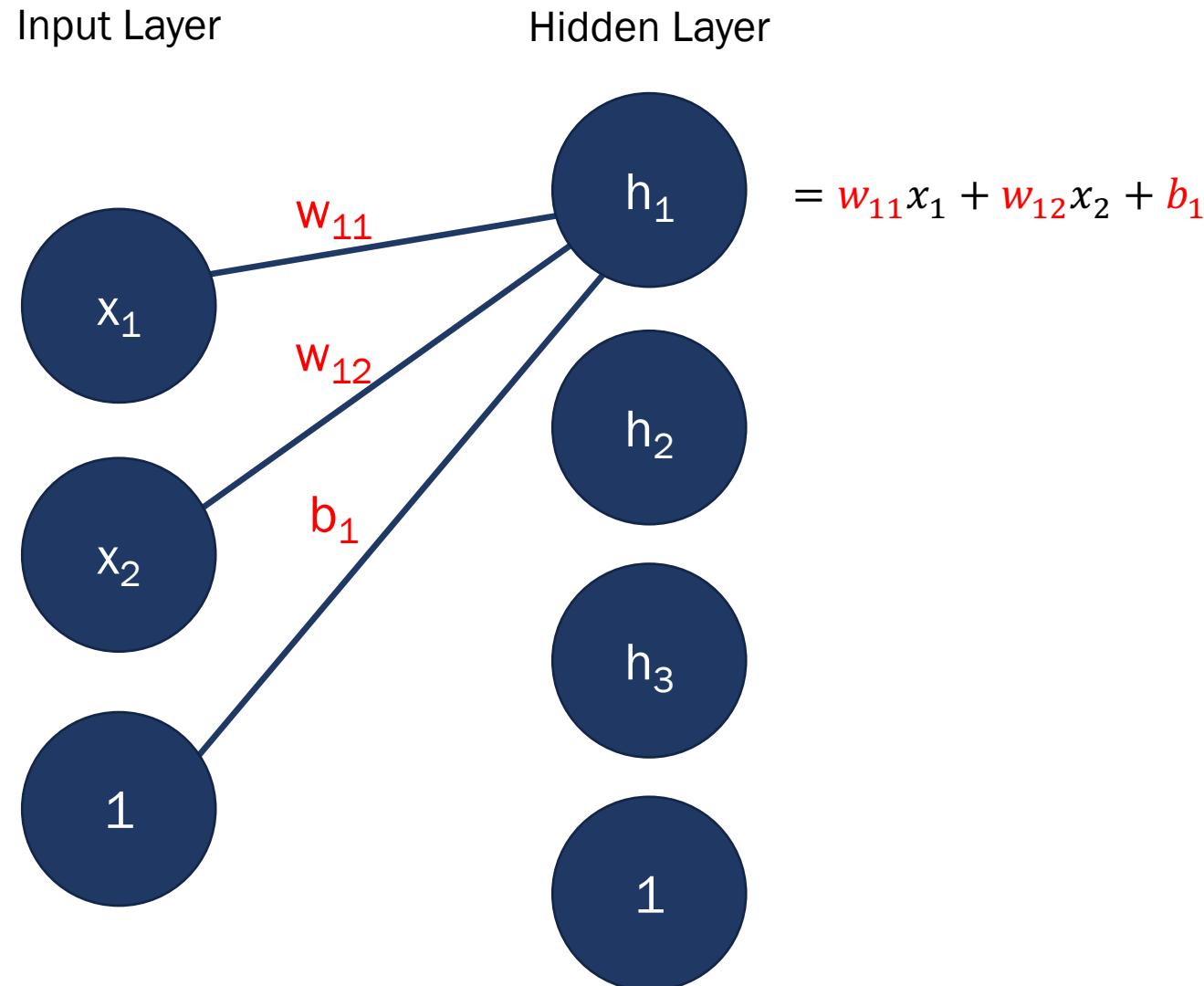
Structure



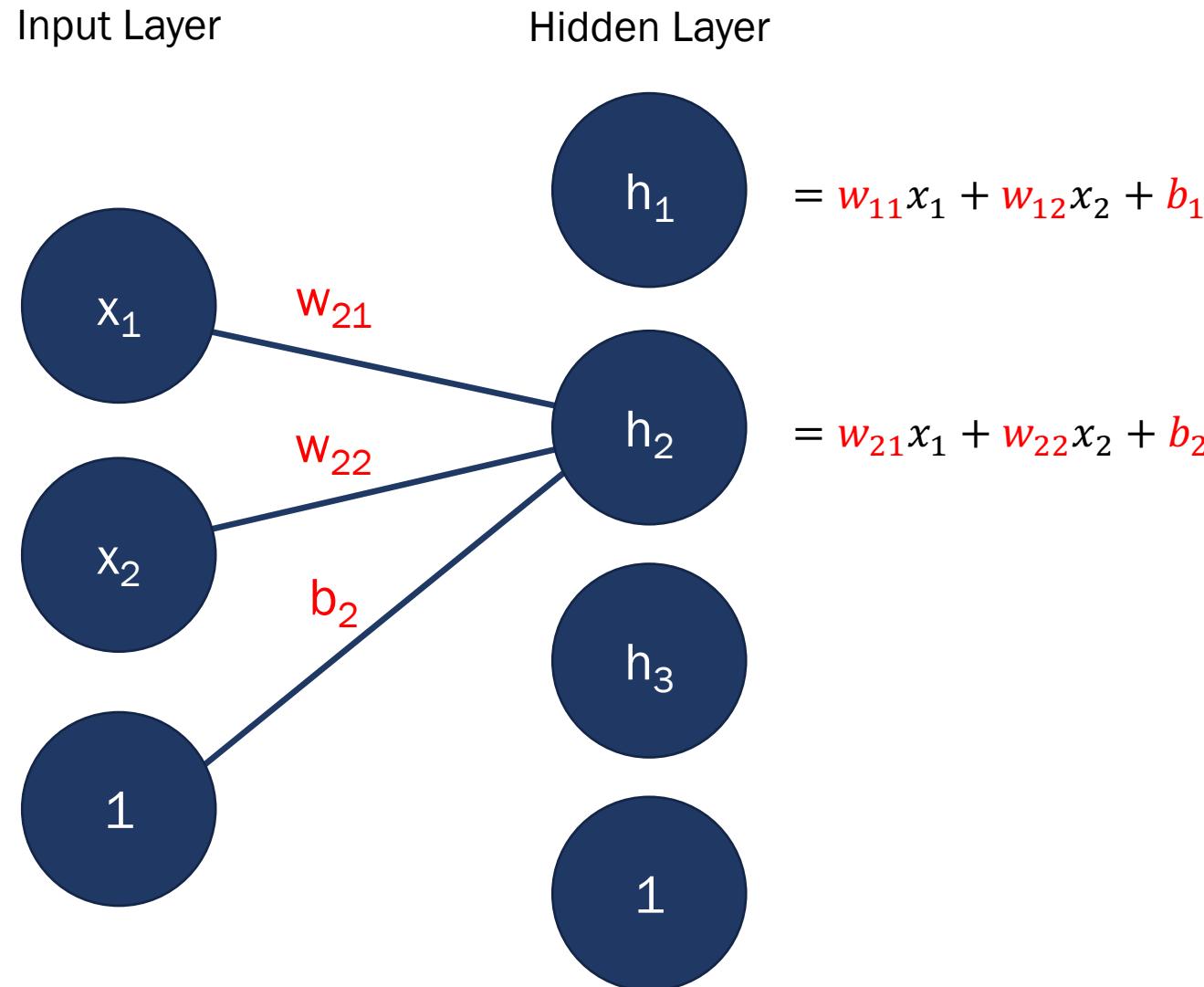
Example



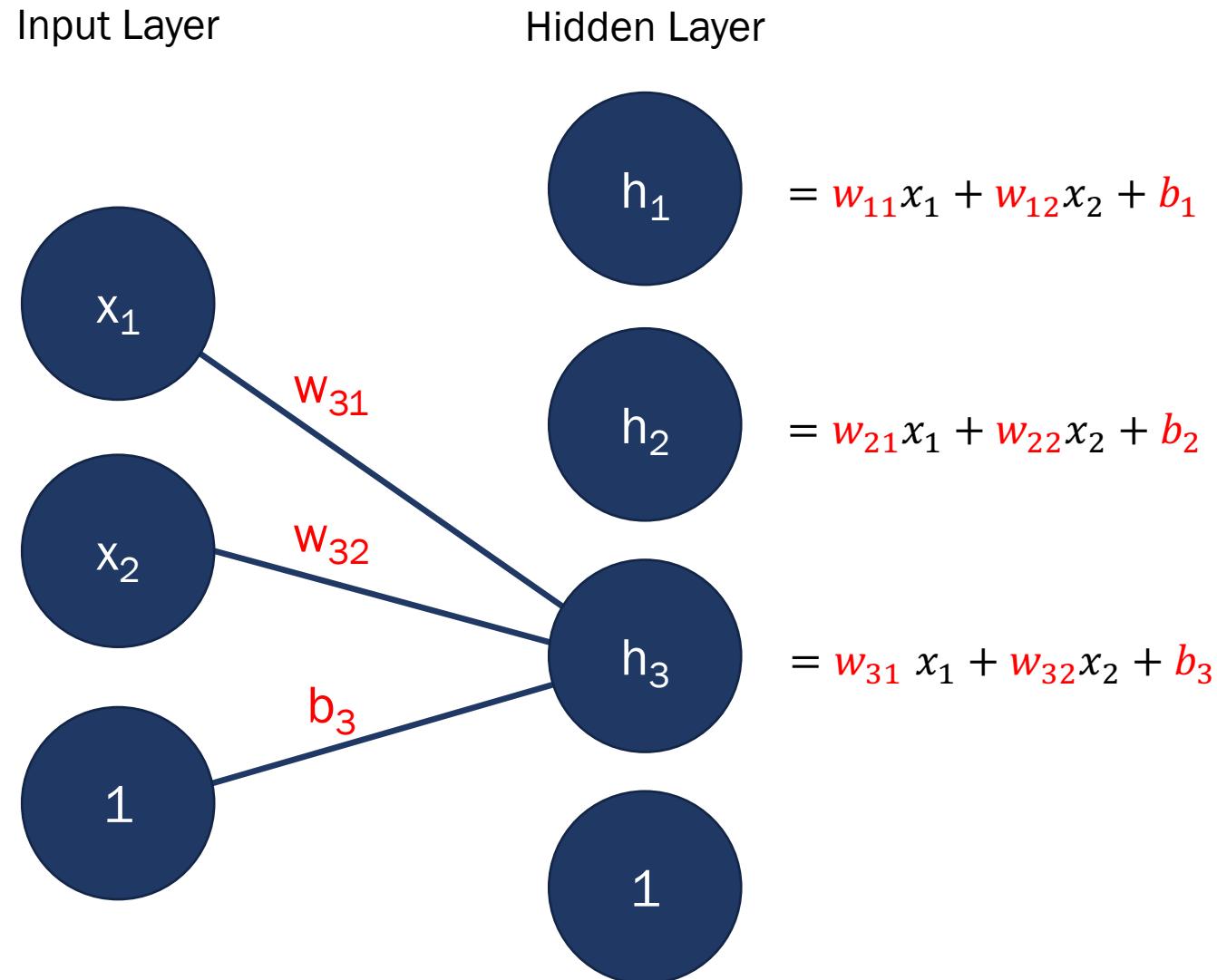
Weighted sum



Weighted sum



Weighted sum



Weighted sum with activation

Input Layer

 x_1 x_2 1

Hidden Layer

 $\sigma(h_1)$ $\sigma(h_2)$ $\sigma(h_3)$ 1 w_{31} w_{32} b_3

$$= \sigma(w_{11}x_1 + w_{12}x_2 + b_1)$$

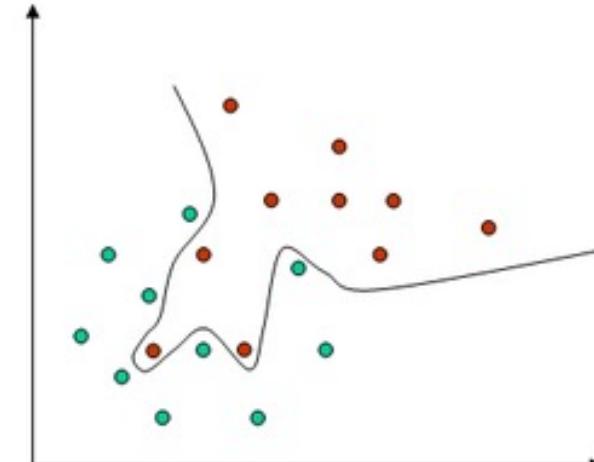
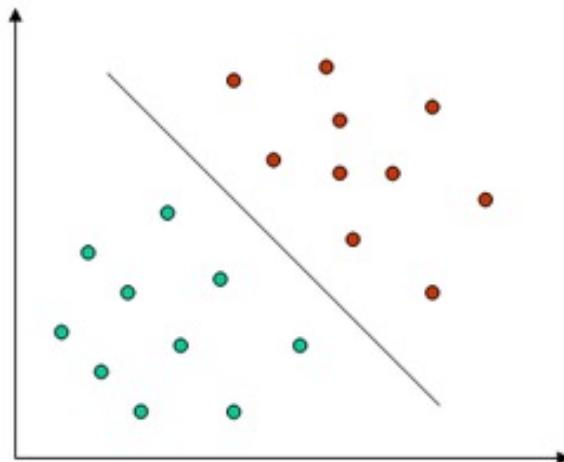
$$= \sigma(w_{21}x_1 + w_{22}x_2 + b_2)$$

$$= \sigma(w_{31}x_1 + w_{32}x_2 + b_3)$$

Why Activationfunction?

Without nonlinear activation, neural networks are just linear regression!!!

- Without an activation function, the neural network would not capture nonlinear, complex relationships
- Via the activation function one can determine essential properties of the model

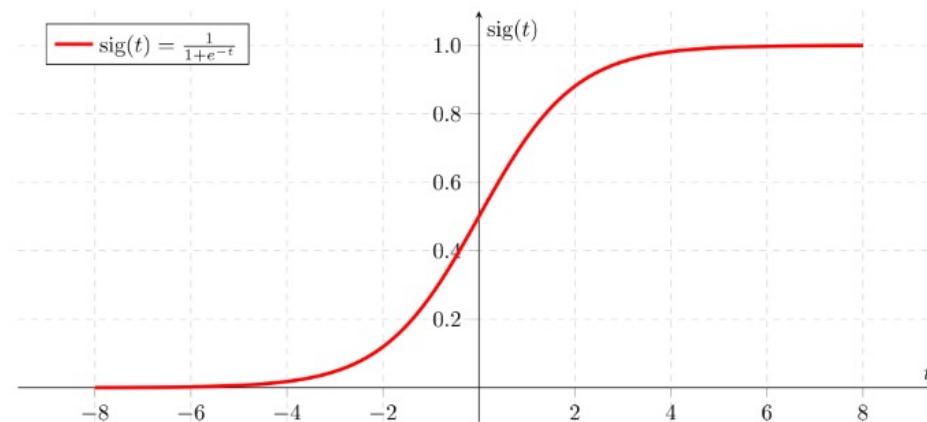


Activationfunction: Example

Different activation functions come with pros and cons. E.g. the sigmoid function is very computationally inefficient (exponential!) while ReLU is very efficient but loses probabilistic interpretation

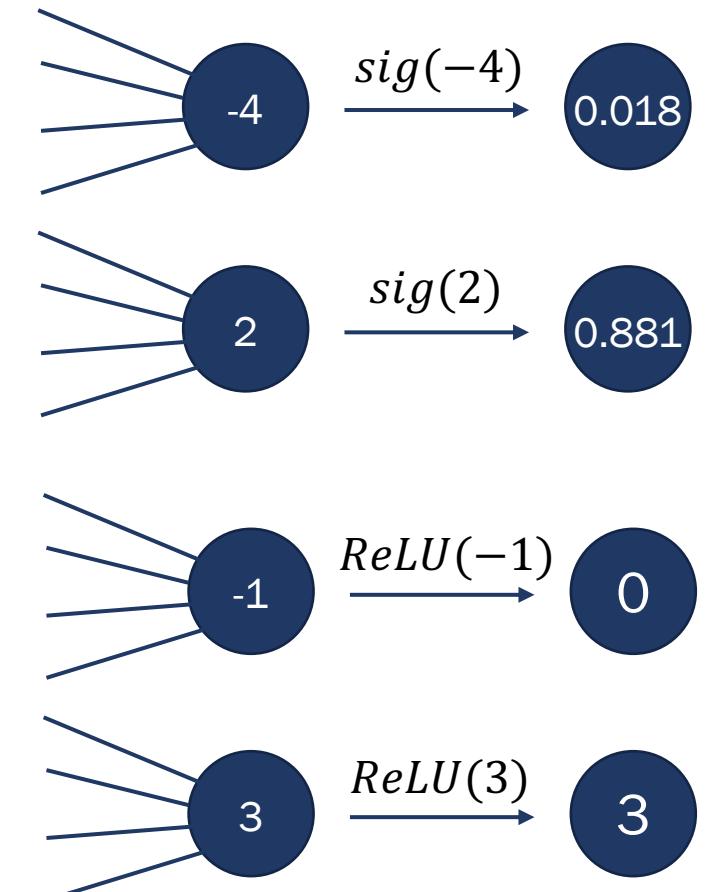
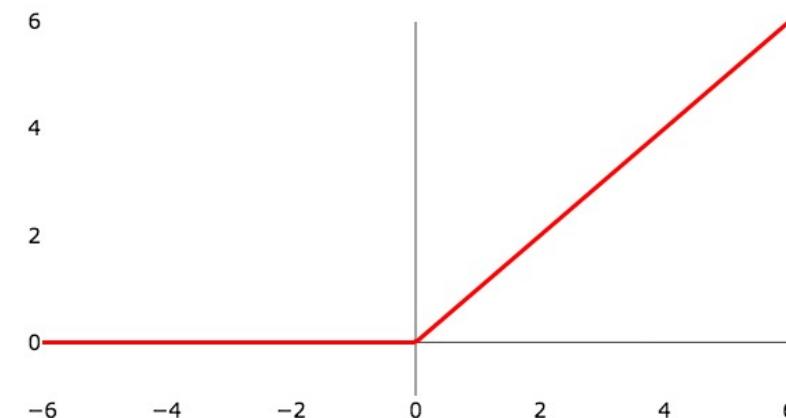
Sigmoid

- Values between 0 and 1
- Mainly for classification
- Mostly used in **output layer**

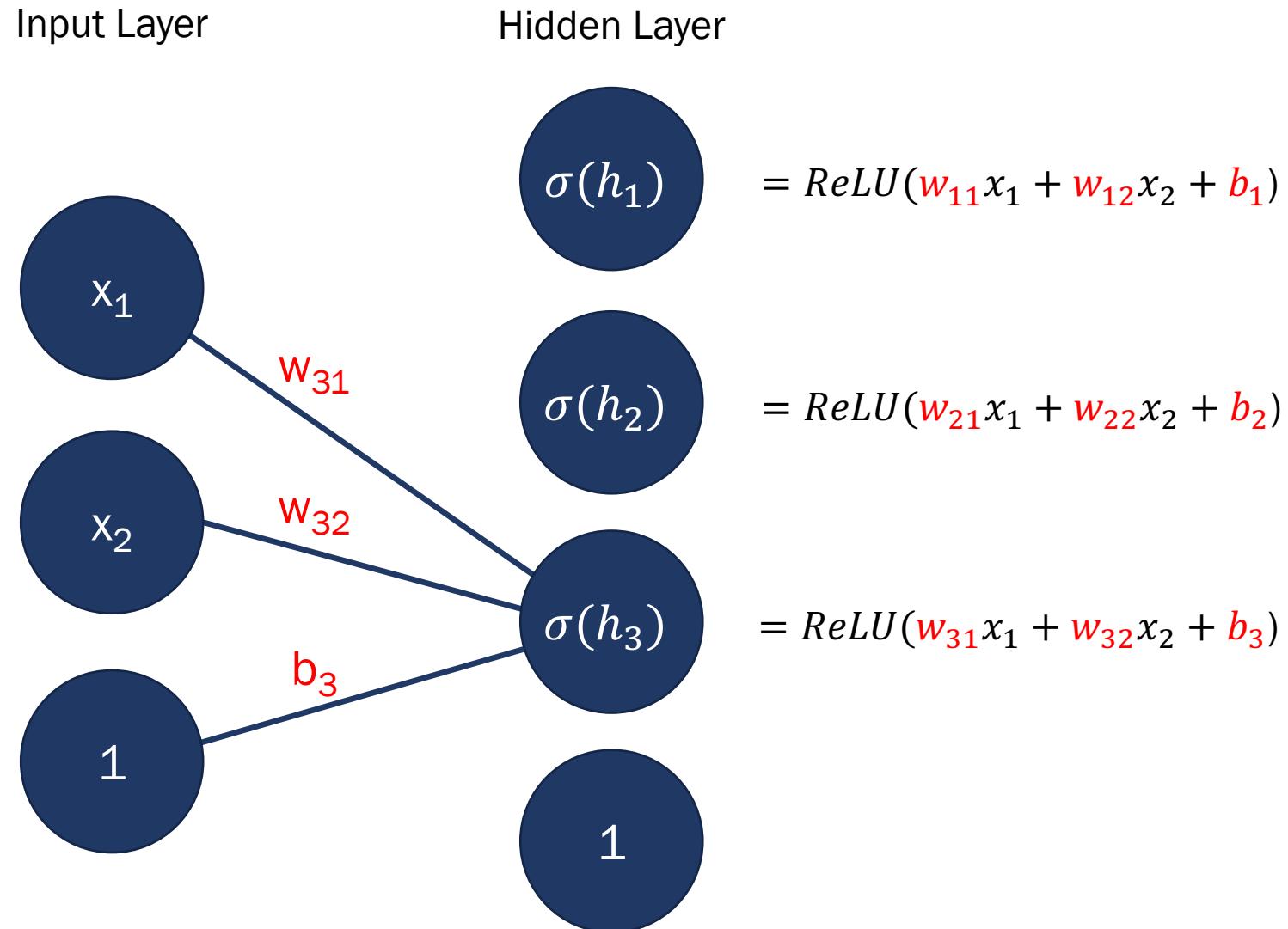


$$\text{ReLU}(t) = \max(0, t)$$

- Negative values are set to zero
- Mostly used in **hidden layers**

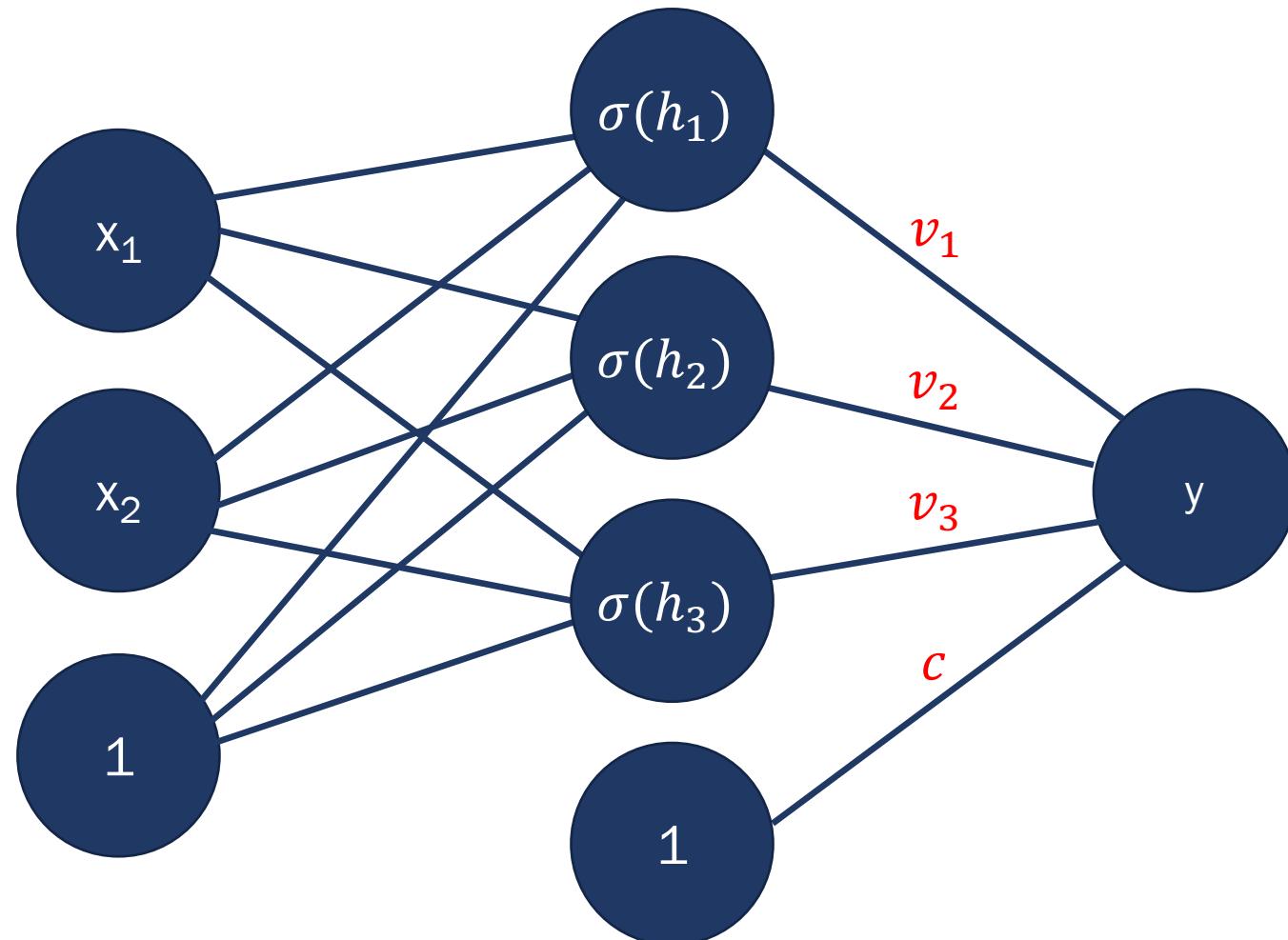


Weighted sum



Final Structure

Input Layer



$$= v_1 \sigma(h_1) + v_2 \sigma(h_2) + v_3 \sigma(h_3) + c$$

The most important step is still missing

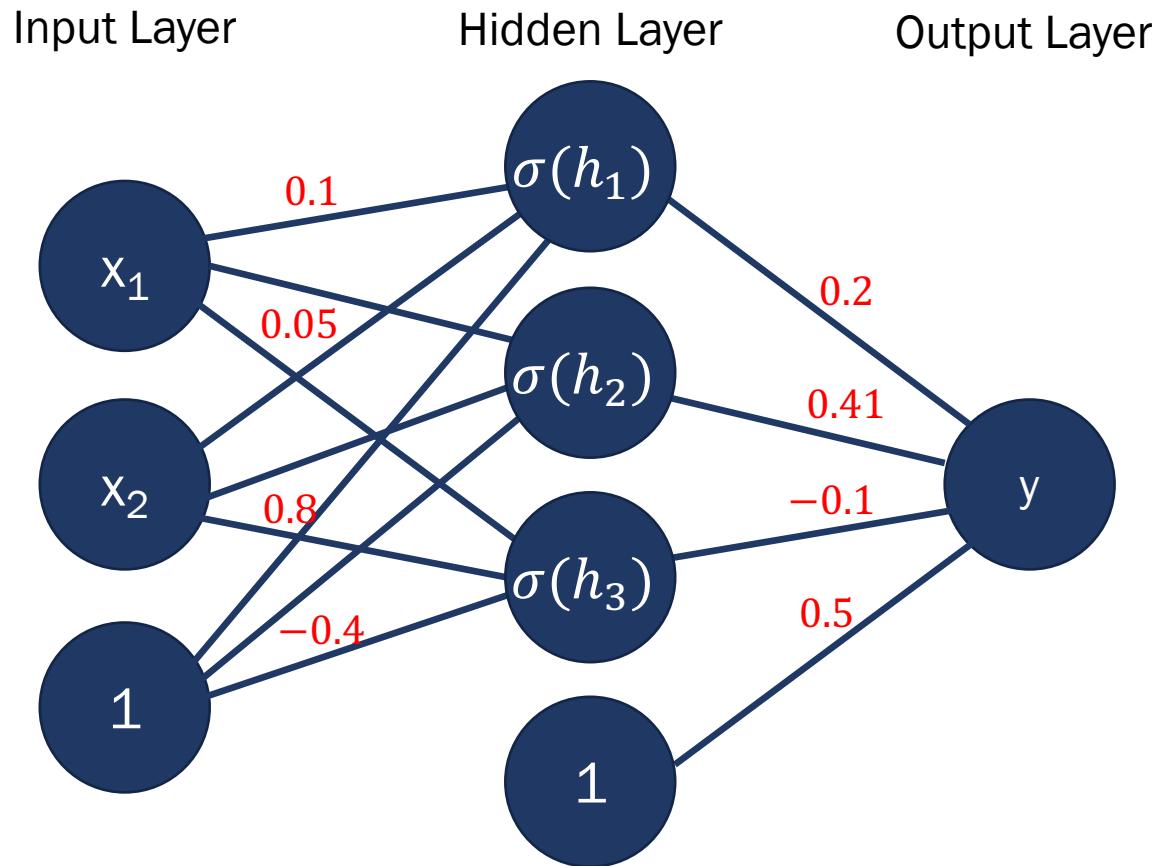
Do you have an idea?

Goal: Fit the weights

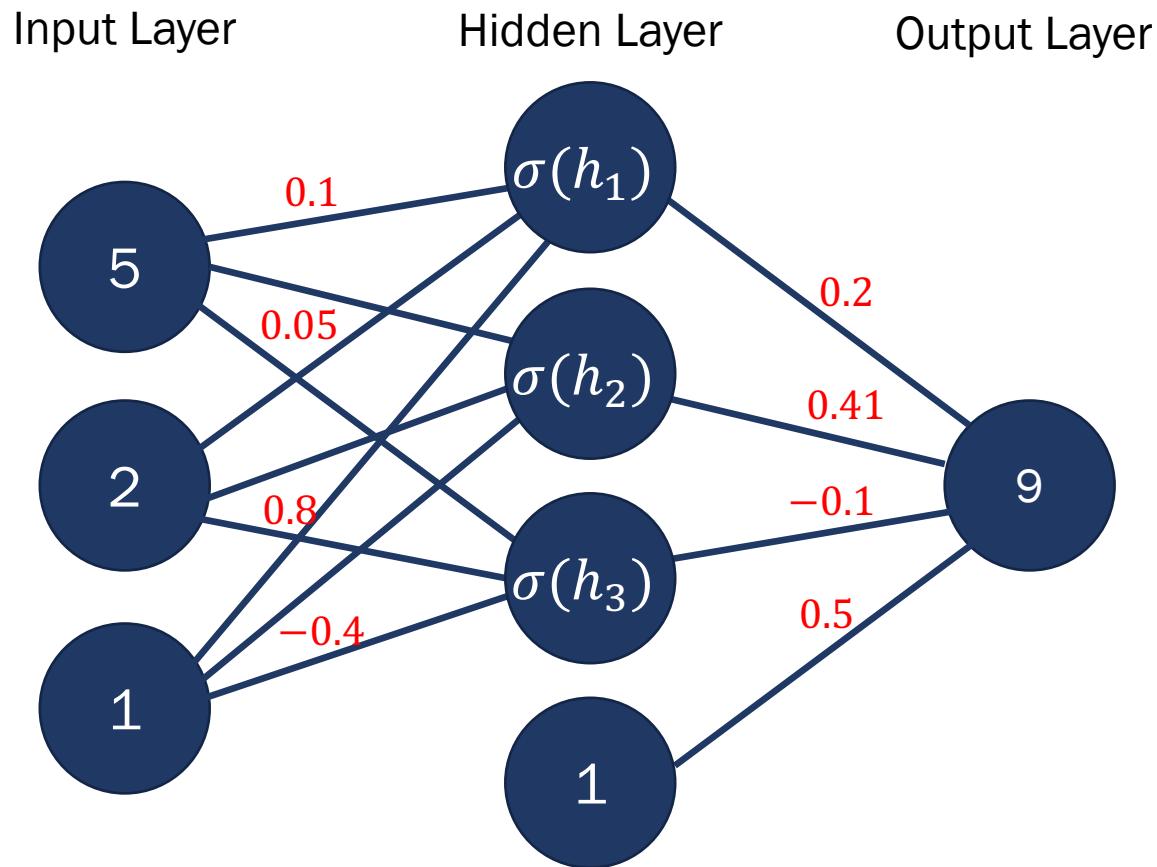
Train a neural network = fit the weights to the data.



Initialize weights randomly



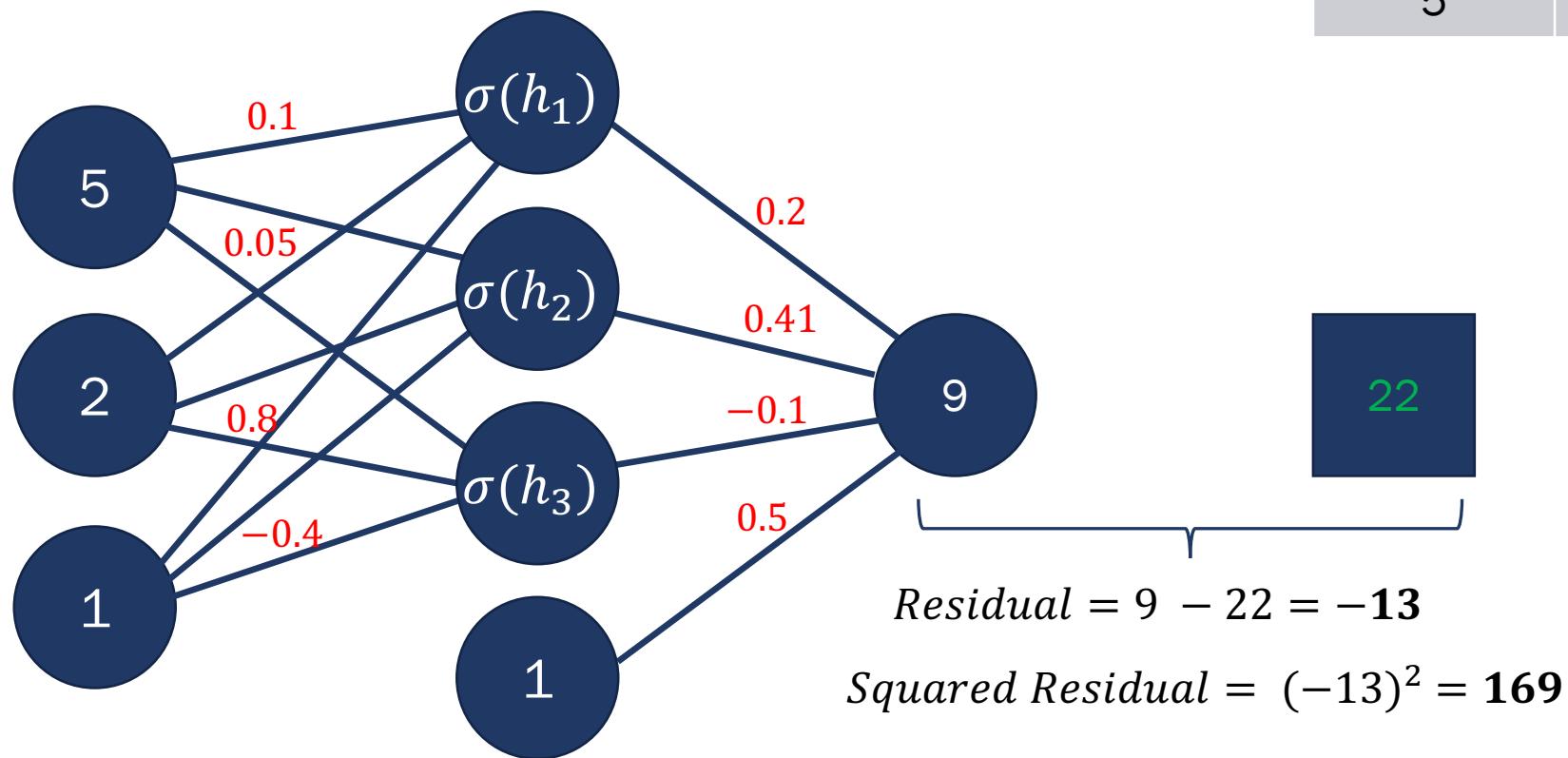
Insert data point



x_1	x_2	\hat{y}
5	2	22

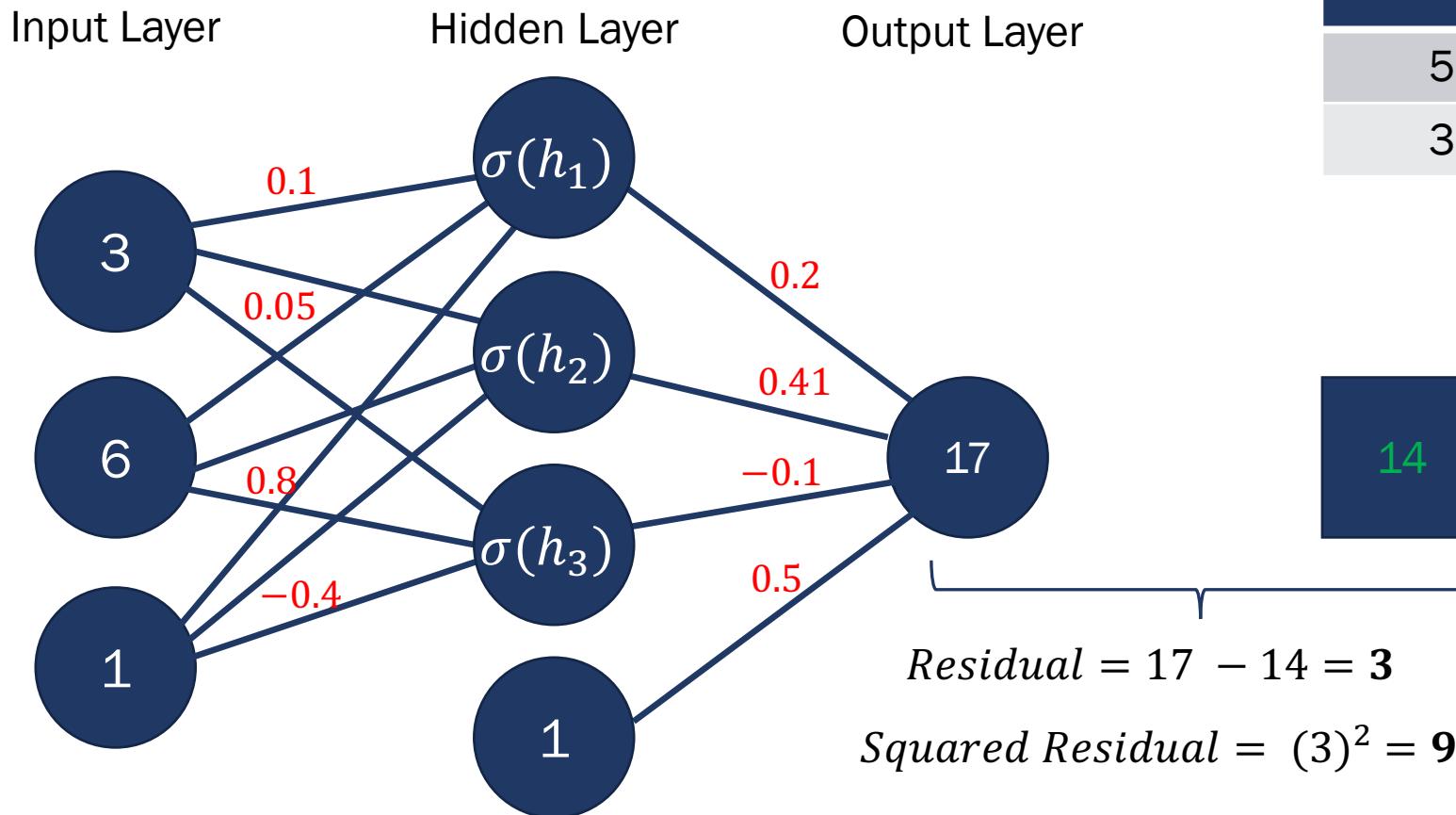
Compare prediction and label (residuals)

Input Layer Hidden Layer Output Layer



x_1	x_2	\hat{y}
5	2	22

Compare prediction and label (residuals)



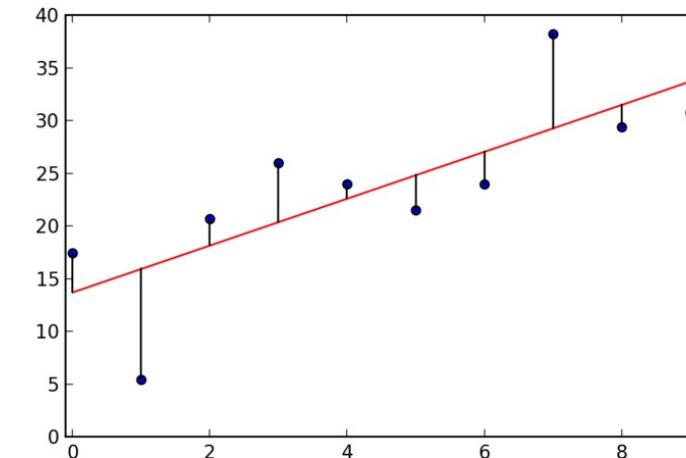
x_1	x_2	\hat{y}
5	2	22
3	6	14

Loss-Function

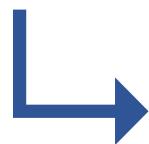
In order to calculate the quality of forecasts, a so-called loss function is required. This serves as a measure for the deviation of the forecasts from the observed values. We use the residual sum of squares.

Sum of squared residuals (SSR)

- Sums the squares of the deviations/residuals
- Squaring makes values non-negative
- Easy to calculate and differentiable



$$SSR = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = (9 - 22)^2 + (17 - 14)^2 + (y_3 - \hat{y}_3)^2 + \dots = 169 + 9 + (y_3 - \hat{y}_3)^2 + \dots$$



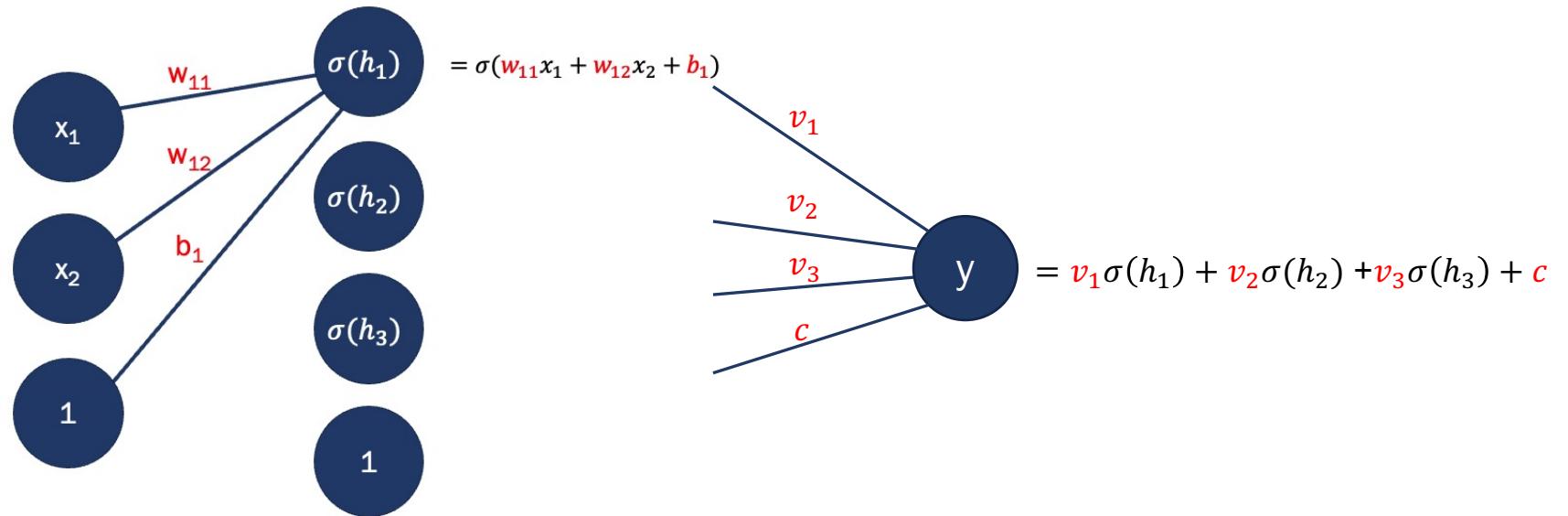
We want to minimize this sum!

Minimize the sum of squared residuals

The goal is to change the weights & constants so that the error sum of squares is minimized.

Mathematically expressed: Approximate the minimum of the loss function

$$\begin{aligned}
 f(w_{11}, \dots, v_1, \dots, b_1, \dots, c) &= (y - \hat{y})^2 \\
 &= (v_1\sigma(h_1) + v_2\sigma(h_2) + v_3\sigma(h_3) + c - \hat{y})^2 \\
 &= (v_1\sigma(w_{11}x_1 + w_{12}x_2 + b_1) + v_2\sigma(w_{21}x_1 + w_{22}x_2 + b_2) + v_3\sigma(w_{31}x_1 + w_{32}x_2 + b_3) + c - \hat{y})^2
 \end{aligned}$$

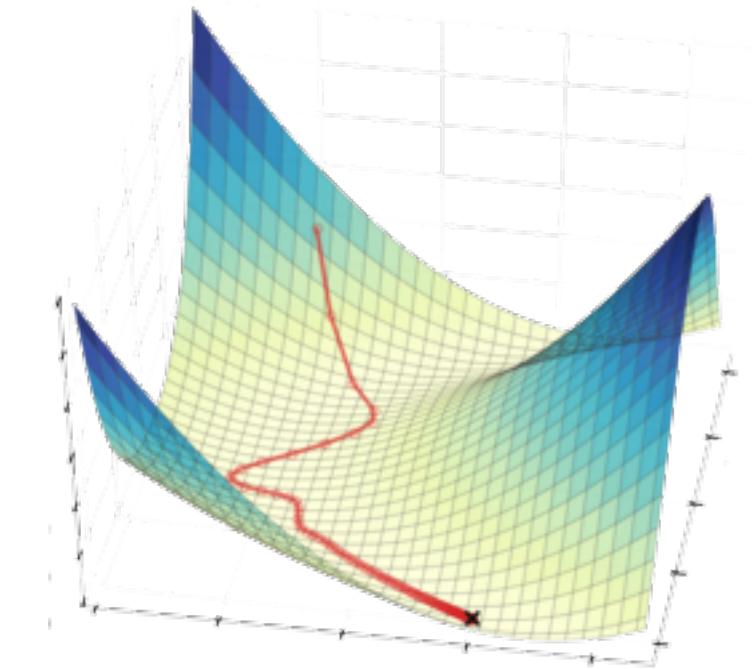
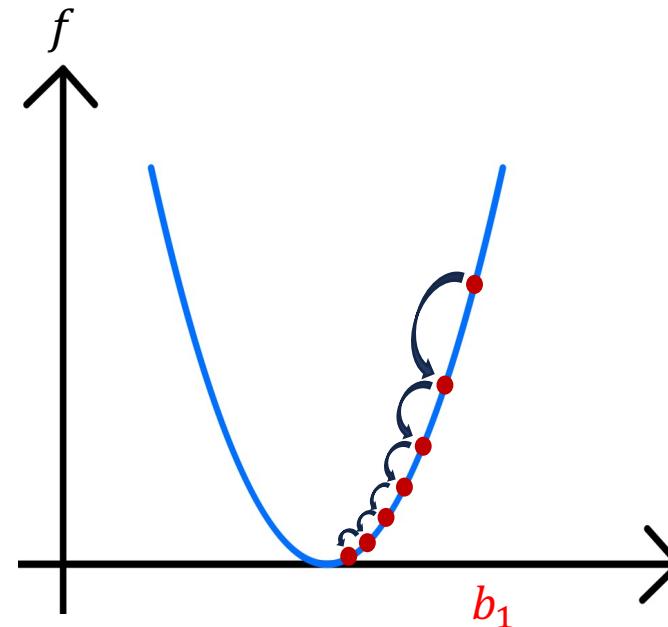
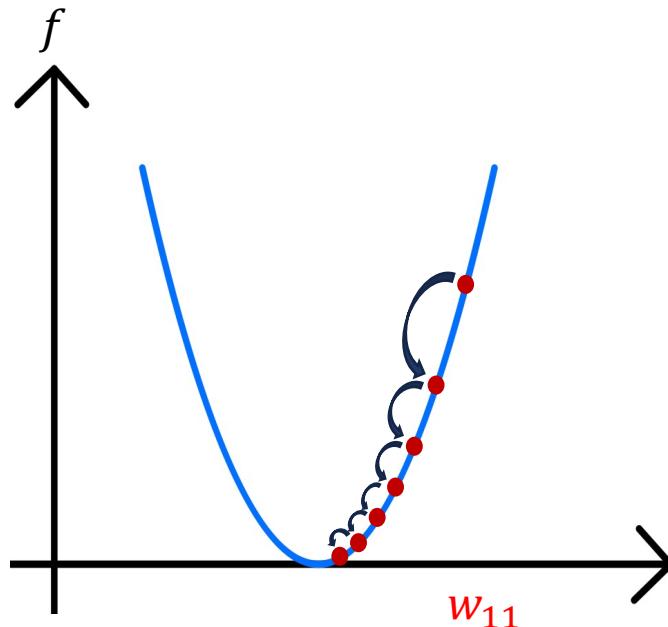


Minimize the sum of squared residuals

The goal is to change the weights & constants so that the error sum of squares is minimized.

Mathematically expressed: Approximate the minimum of the loss function

$$\begin{aligned}
 f(w_{11}, \dots, v_1, \dots, b_1, \dots, c) &= (y - \hat{y})^2 \\
 &= (v_1\sigma(h_1) + v_2\sigma(h_2) + v_3\sigma(h_3) + c - \hat{y})^2 \\
 &= (v_1\sigma(w_{11}x_1 + w_{12}x_2 + b_1) + v_2\sigma(w_{21}x_1 + w_{22}x_2 + b_2) + v_3\sigma(w_{31}x_1 + w_{32}x_2 + b_3) + c - \hat{y})^2
 \end{aligned}$$



Gradient descent

The gradient descent method is one of the best known optimization methods for neural networks. With the gradient method, a **local** minimum of the loss function can be easily approximated.

$$\begin{aligned}
 f(w_{11}, \dots, v_1, \dots, b_1, \dots, c) &= (y - \hat{y})^2 \\
 &= (v_1 \sigma(h_1) + v_2 \sigma(h_2) + v_3 \sigma(h_3) + c - \hat{y})^2 \\
 &= (v_1 \sigma(w_{11}x_1 + w_{12}x_2 + b_1) + v_2 \sigma(w_{21}x_1 + w_{22}x_2 + b_2) + v_3 \sigma(w_{31}x_1 + w_{32}x_2 + b_3) + c - \hat{y})^2
 \end{aligned}$$

Chain rule gives...

...for a data point: $\frac{\partial f(w_{11}, \dots)}{\partial (w_{11})} = \frac{\partial (y - \hat{y})^2}{\partial (w_{11})} = 2 (y - \hat{y}) \cdot \frac{\partial y}{\partial (w_{11})} = 2 (y - \hat{y}) \cdot v_1 \cdot \frac{\partial \sigma(h_1)}{\partial (w_{11})} \cdot x_1$

...for the whole dataset: $\frac{\partial SSR}{\partial (w_{11})} = \frac{\partial}{\partial (w_{11})} \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \sum_{i=1}^n \frac{\partial (y_i - \hat{y}_i)^2}{\partial (w_{11})} = \sum_{i=1}^n \left[2 (y_i - \hat{y}_i) \cdot v_1 \cdot \frac{\partial \sigma(h_1^i)}{\partial (w_{11})} \cdot x_1 \right]$

Or in general: $\frac{\partial SSR}{\partial (\theta_j)} = \sum_{i=1}^n \left[2 (y_i - \hat{y}_i) \cdot \frac{\partial y_i}{\partial (\theta_j)} \right]$

Learning Rate

Now the "learning" or the "training" begins. All parameters (=weights and constants) must be adjusted with the help of the derivation.

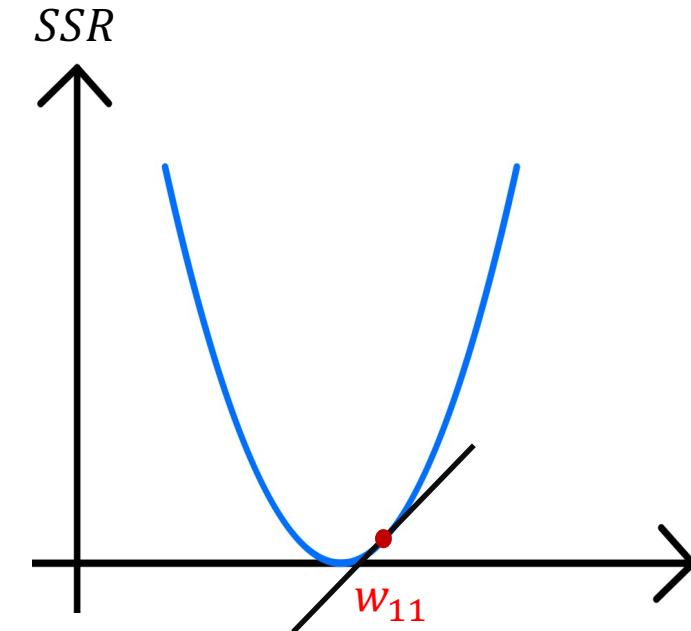
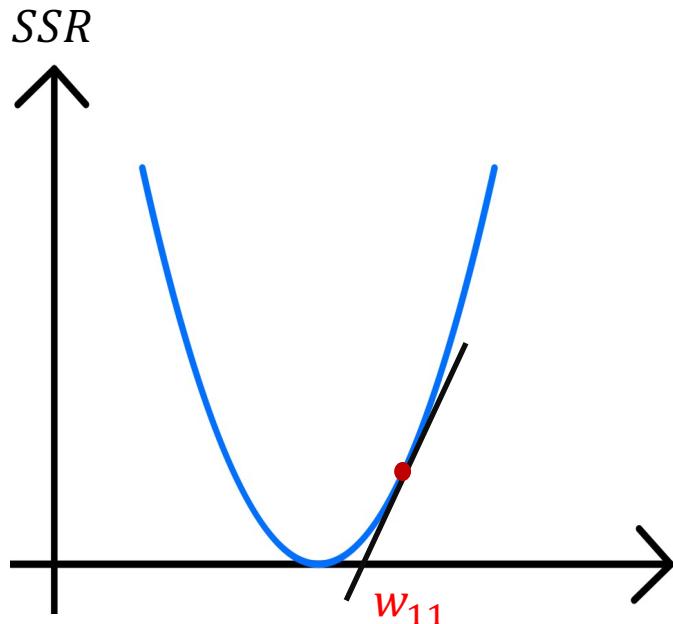
The most important step:

$$\text{Set } w_{11} = w_{11} - \alpha \cdot SSR'(\mathbf{w}_{11}, \dots)$$

$$\text{In general: } \theta_j = \theta_j - \alpha \cdot \frac{\partial SSR}{\partial (\theta_j)}$$

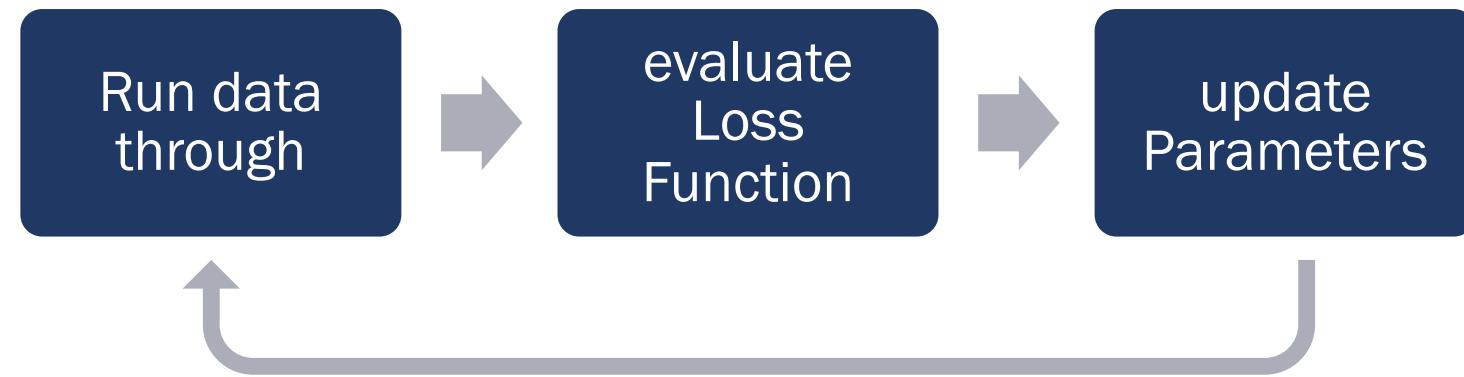
Learning Rate α

- The learning rate indicates how much the parameters have to be corrected in each run
- E. g. $\alpha = 0.05$



Process of Learning

Now the "learning" or the "training" begins. All parameters (=weights and constants) must be adjusted with the help of the derivation.



x_1	x_2	y	\hat{y}
5	2	9	22
3	6	17	14

x_1	x_2	y	\hat{y}
5	2	17.7	22
3	6	15.2	14

x_1	x_2	y	\hat{y}
5	2	21.3	22
3	6	14.5	14

Training & Testing

To train a model, one must use training data.

To evaluate the model, one must use the so-called test data. With them it is possible to check whether the model is reliable also on foreign data.

x_1	x_2	y	\hat{y}
5	2	21.3	22
3	6	14.5	14
7	-2	4.6	4
0.5	3	9.7	10

Training Data
Accuracy: 95%

- When you receive a data set, you must always split it into a training and a test data set
- Test data must not be used for training
- The model usually performs better on the training data than on the test data.

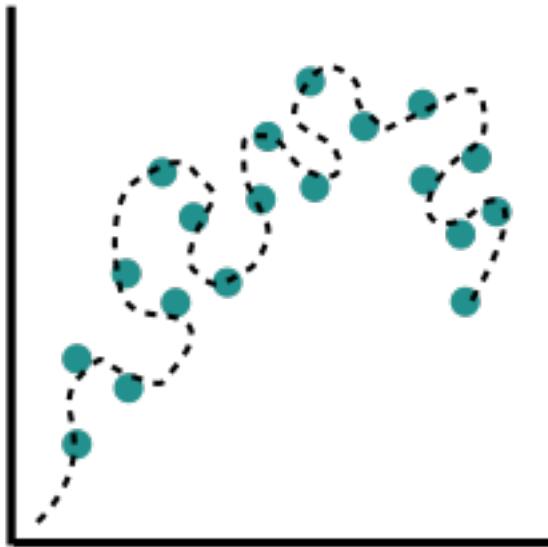
x_1	x_2	y	\hat{y}
2	1	4.4	6
9	3	11.4	12.9

Test Data
Accuracy: 92%

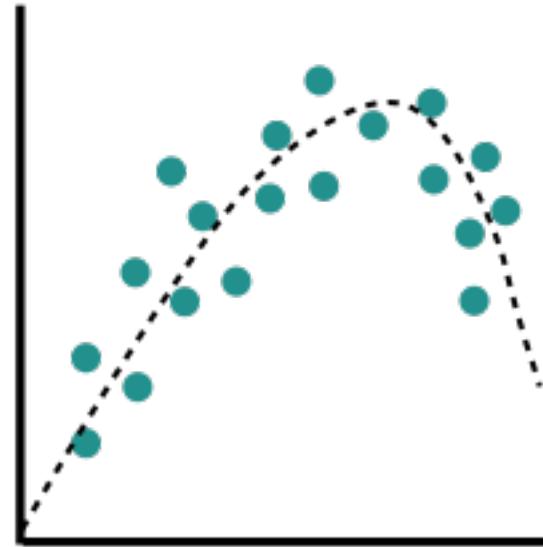
Overfitting vs. Underfitting

If you fit the model too much to the data, an overfit may take place. There are several methods to avoid this.

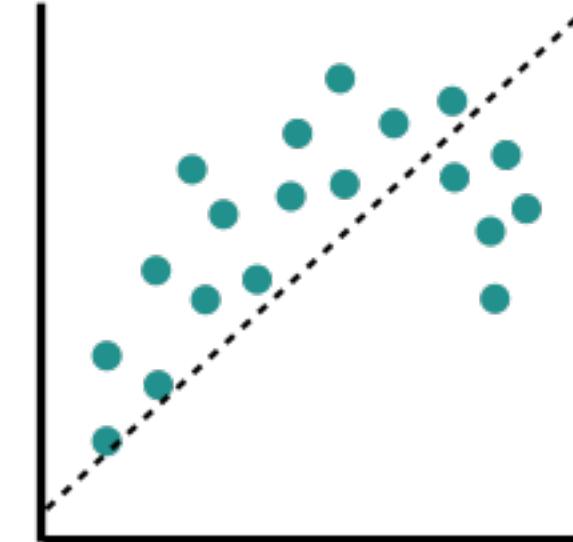
Overfit (Variance)



Robust



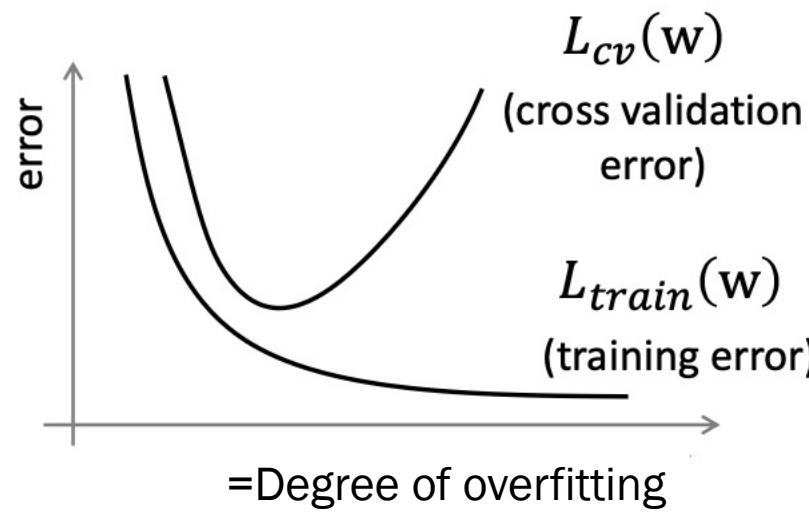
Underfit (Bias)



Overfitting vs. Underfitting: Data Set size

If you fit the model too much to the data, an overfit may take place. There are several methods to avoid this.

Problem



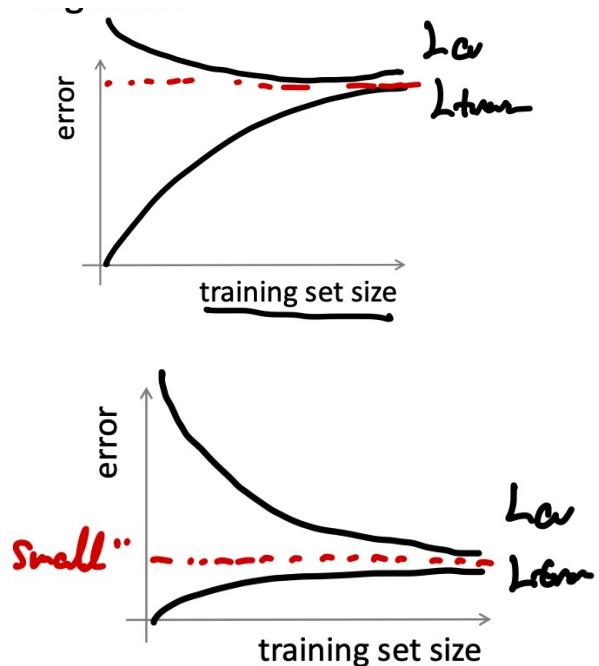
Bias (underfitting): L_{train} will be large and $L_{cv} \approx L_{train}$

Variance (overfitting): L_{train} will be low and $L_{cv} \gg L_{train}$

Solution

High Bias: getting more training data will **not** (by itself) help much

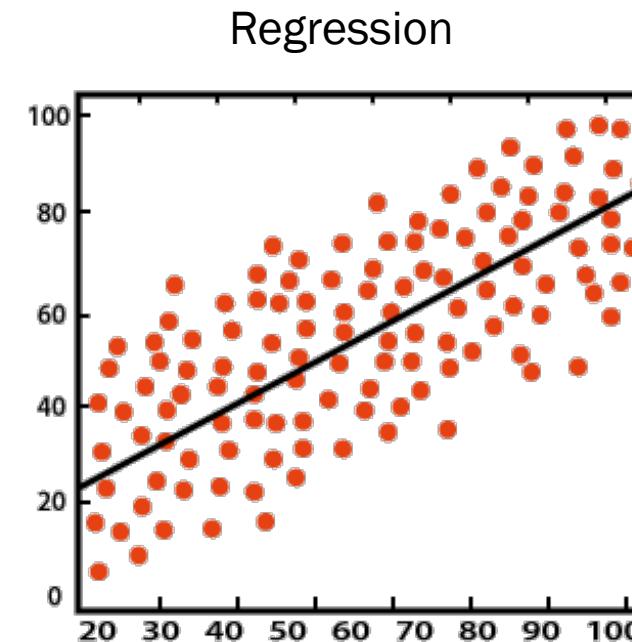
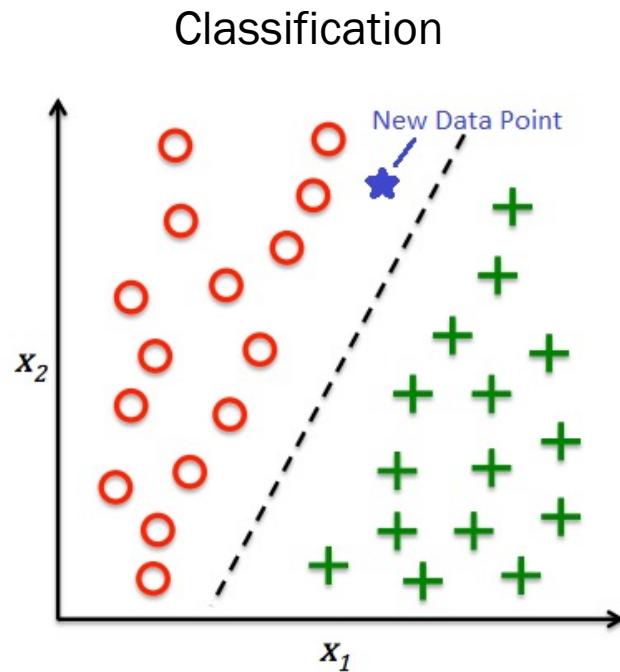
High Variance: getting more training data **may be likely to help**



Deep Neural Networks are very large models, but overfitting can be addressed by using large data sets

Classification vs. Regression

Neural networks are divided into classification and regression models. Which of the two one chooses, depends on the facts.



But...

What are possible disadvantages of neural networks?

Agenda

Intro

How do neural Networks work?

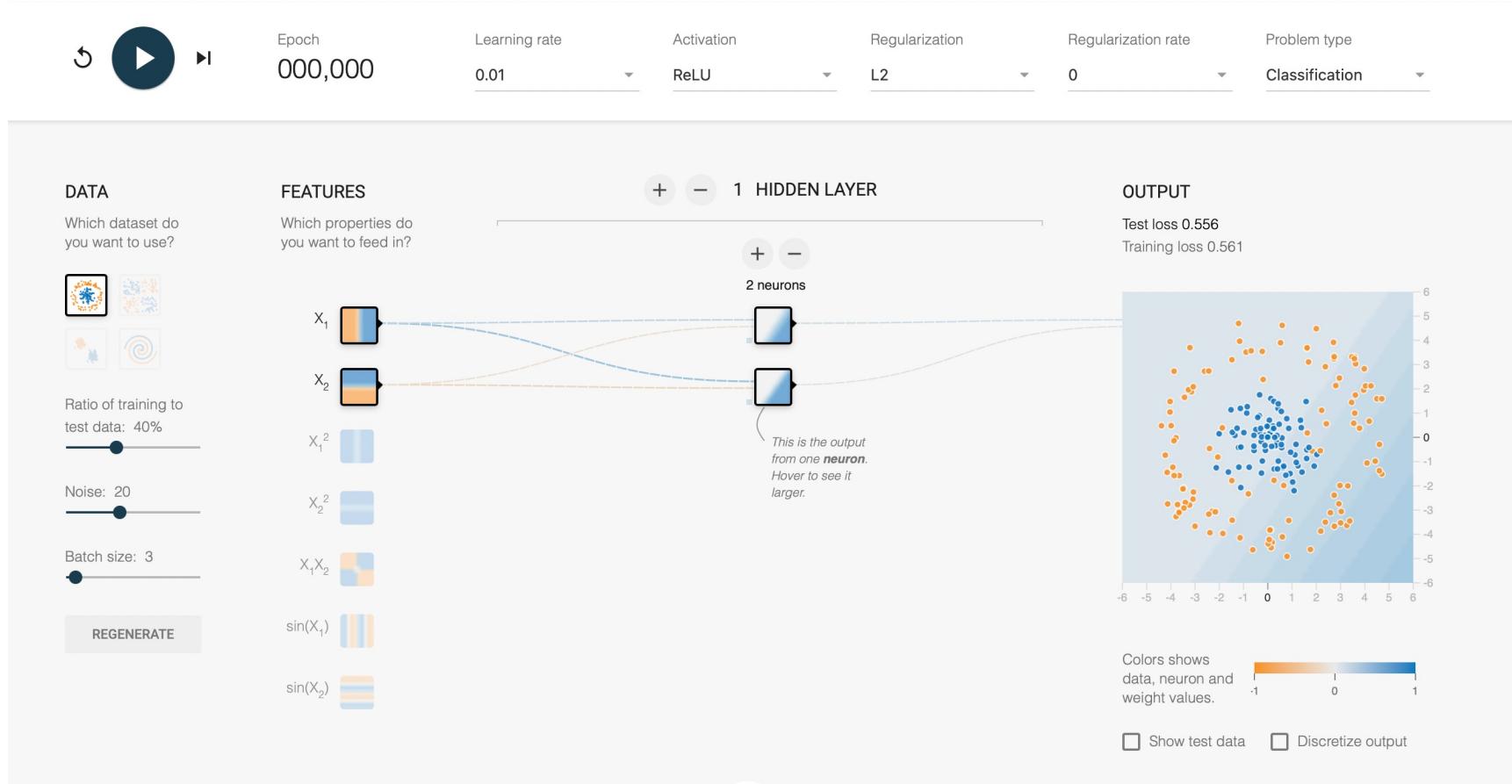


Practical implementation



STADS

Playing around with neural networks...



playground.tensorflow.org