Python AI: How to Build a Neural Network & Make Predictions

The Goal of Artificial Intelligence

- Make predictions

$$2 + 2 = 5$$

$$2 + 2 = 4$$

The Goal of Artificial Intelligence

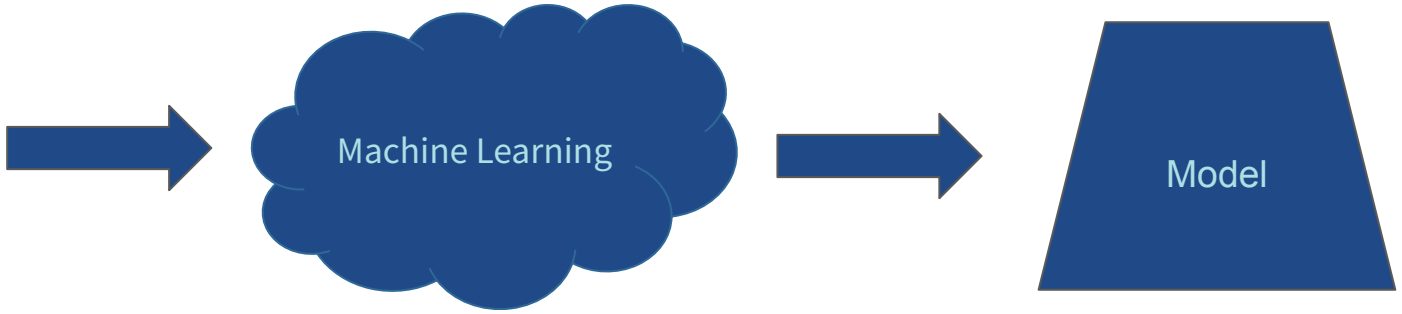
- Make predictions, that mimic human response
- Human response is not the same as human intelligence

Predicting the Sum

```
def aisum(a, b):
    if a == 0: return b
    if b == 0: return a
    if a == 1:
        if b == 1:
            return 2
        elif b == 2:
            return 3
    elif a == 2:
        if b == 1:
            return 3
        elif b == 2:
             return 4
```

Machine Learning

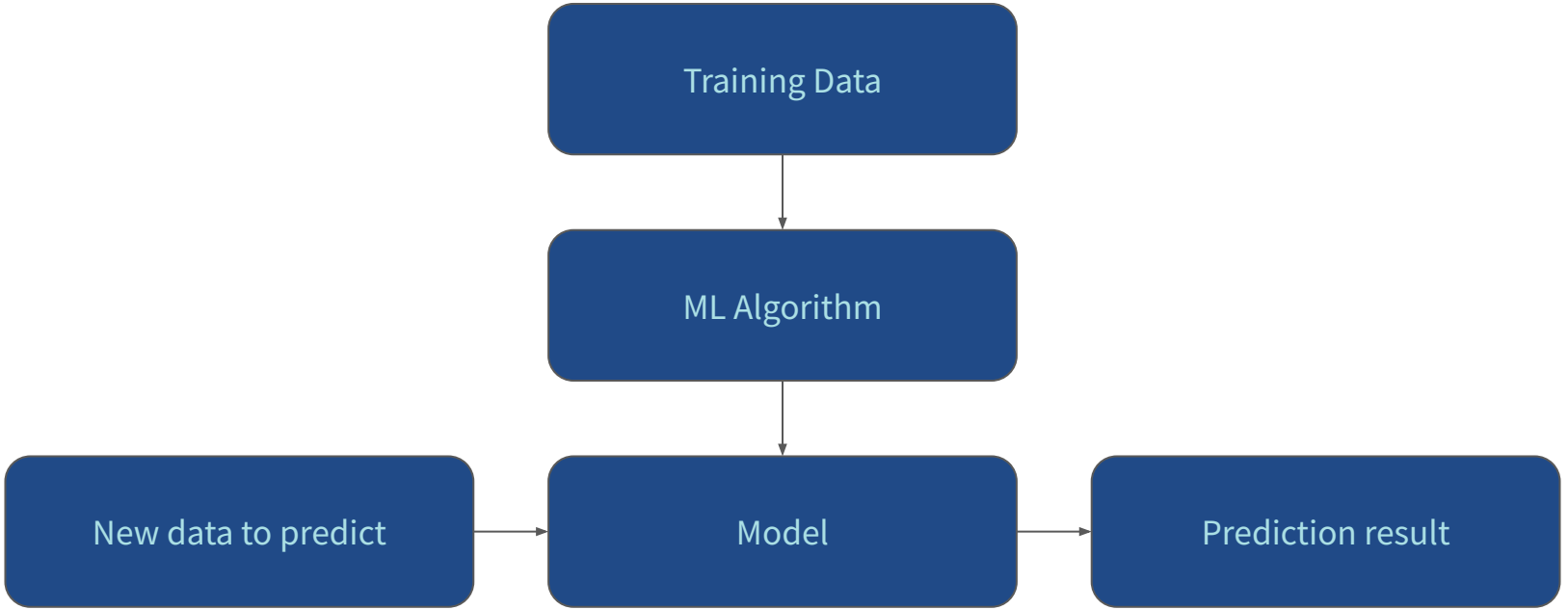
1	0	1
1	1	2
1	2	3
2	0	2
2	2	4



The Goal of Machine Learning

- Make predictions, that mimic human response
- Human response is not the same as human intelligence
- Without being explicitly programmed

Machine Learning



Feature Engineering

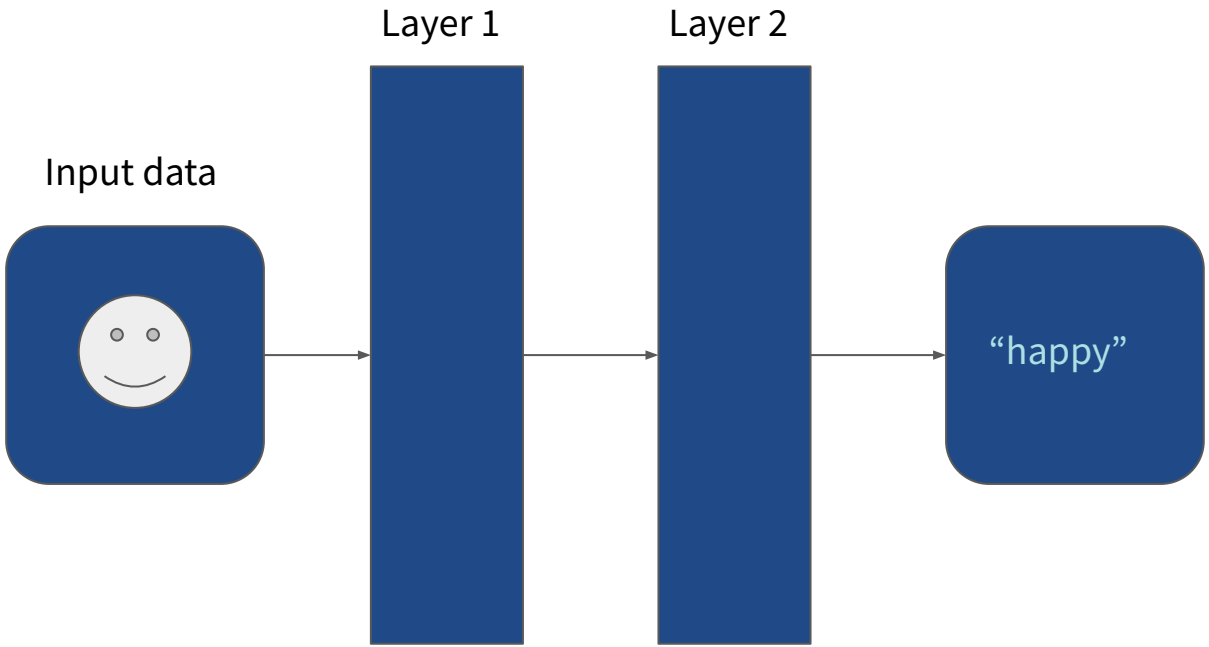
The Goal of Machine Learning

- Make predictions, that mimic human response
- Human response is not the same as human intelligence
- Without being explicitly programmed
- Deep learning applies neural networks to machine learning
- Automates feature engineering

Neural Networks

- Values arranged into layers
- Each layer manipulates the training data
- Output of a layer is the input to the next layer
- The layers are steps in feature engineering

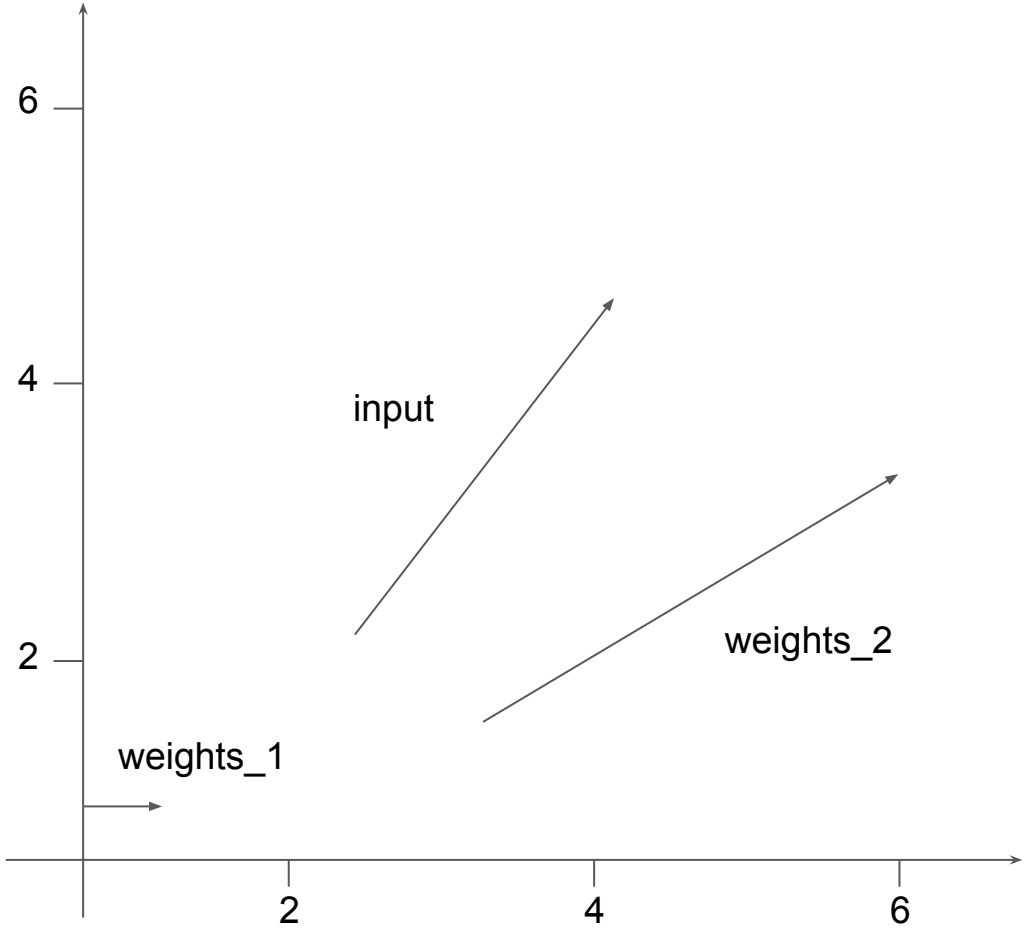
Neural Networks



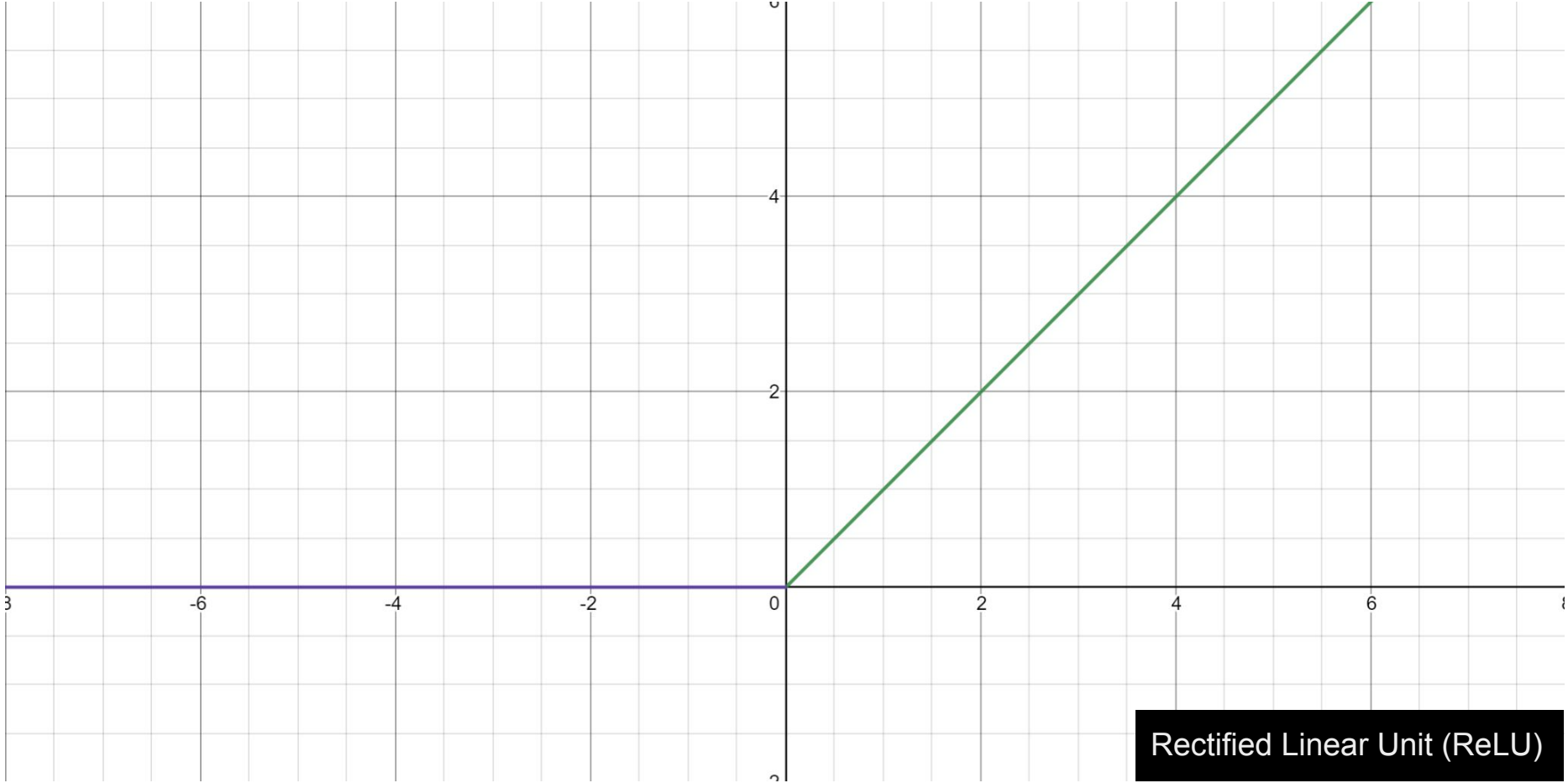
Neural Networks

- Values arranged into layers
- Each layer manipulates the training data
- Output of a layer is the input to the next layer
- The layers are steps in feature engineering
- Vectors
  - numpy ndarray
- Linear regression
  - model the relationship between variables

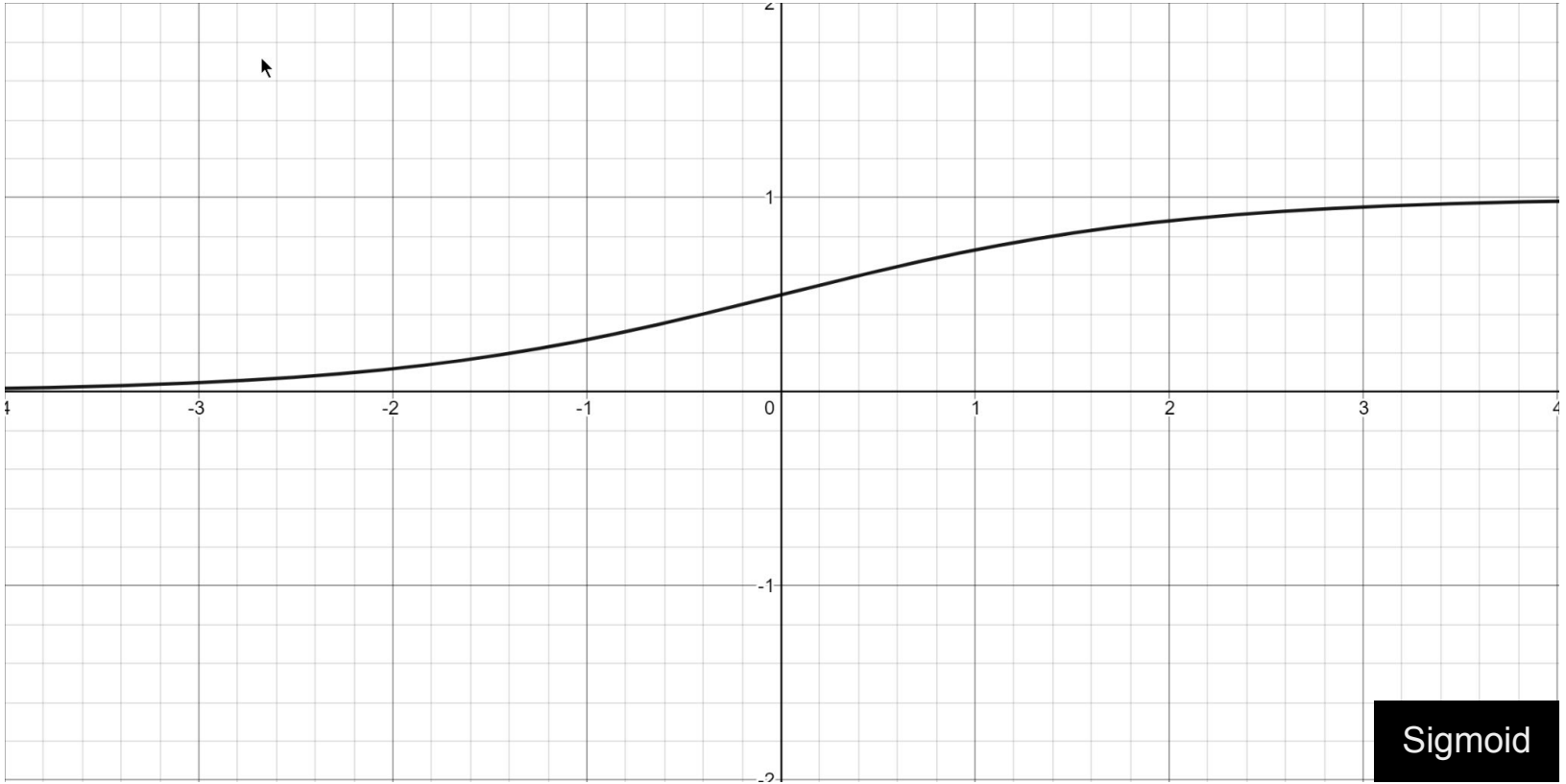
```
price = (weights_area * area) + (weights_age * age) + bias
```

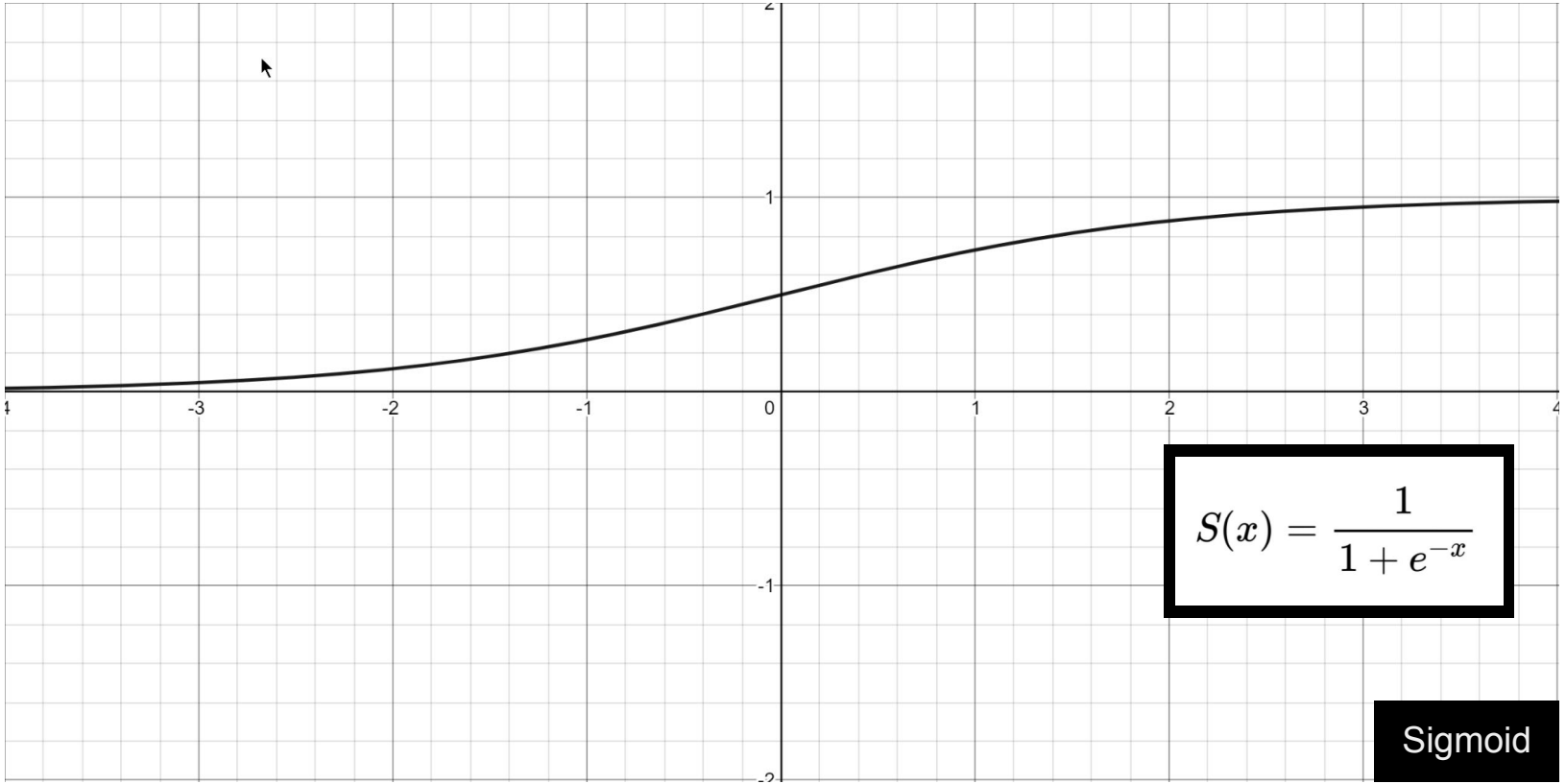

6
4
2
input
weights_2
weights_1
2
4
6

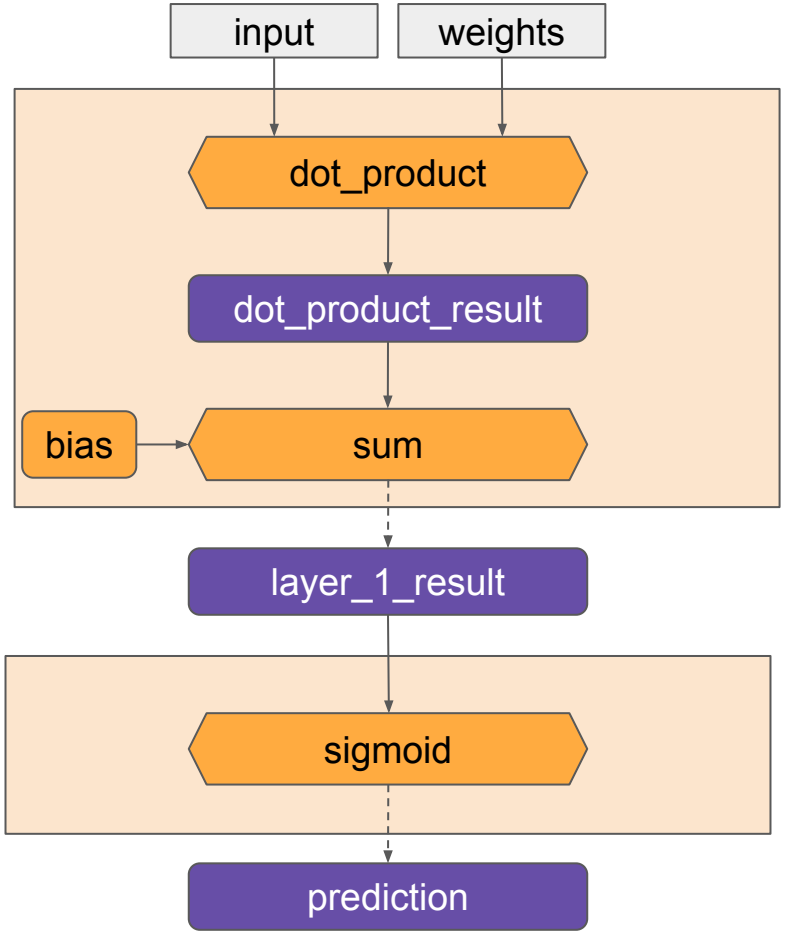
Input Vector	Target
`[1.66, 1.56]`	`1`
`[2, 1.5]`	`0`



Rectified Linear Unit (ReLU)

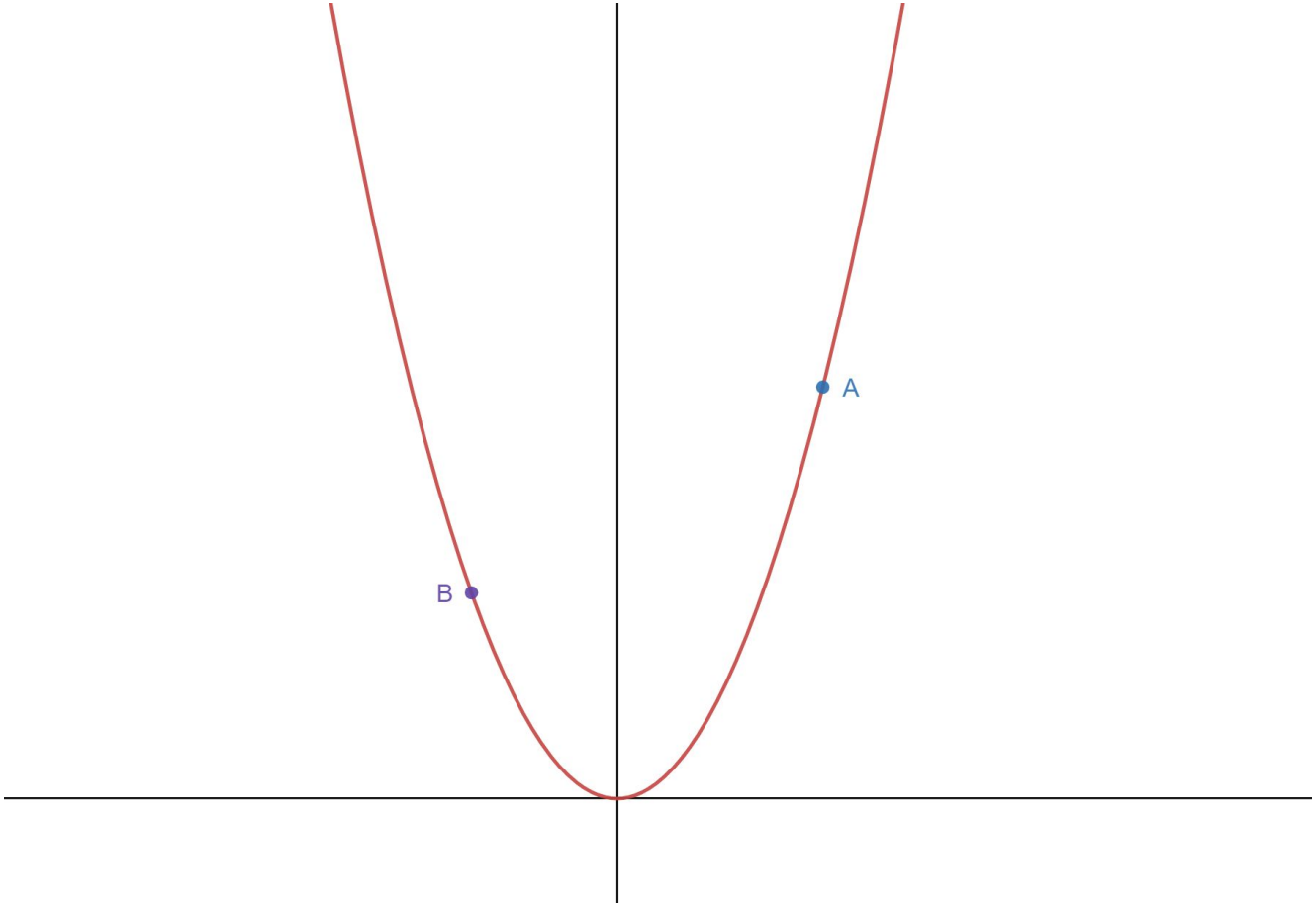

1
-3
-2
-1
0
1
2
3
-1
Sigmoid


2
1
-1
-2
-3
-2
-1
0
1
2
3
4
$S(x) = \frac{1}{1 + e^{-x}}$
Sigmoid


input
weights
dot_product
dot_product_result
bias
sum
layer_1_result
sigmoid
prediction

How Good Is The Prediction?

- Cost or loss function
- Simple mean squared error (MSE)
- $(\text{prediction} - \text{target})^2$


A
B

Computing the Derivative

- Use the derivative to update the values in the network
- Derivative is also called the ‘gradient’
- Adjust the weights with the ‘gradient descent’ process

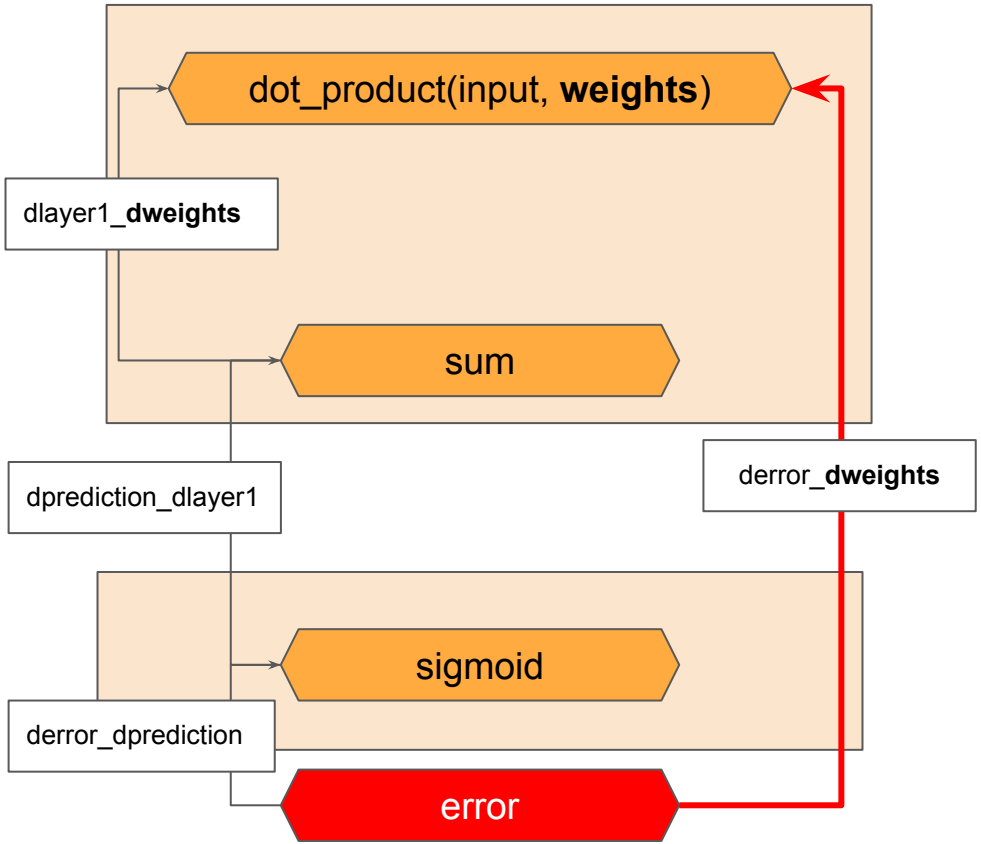
Power Rule

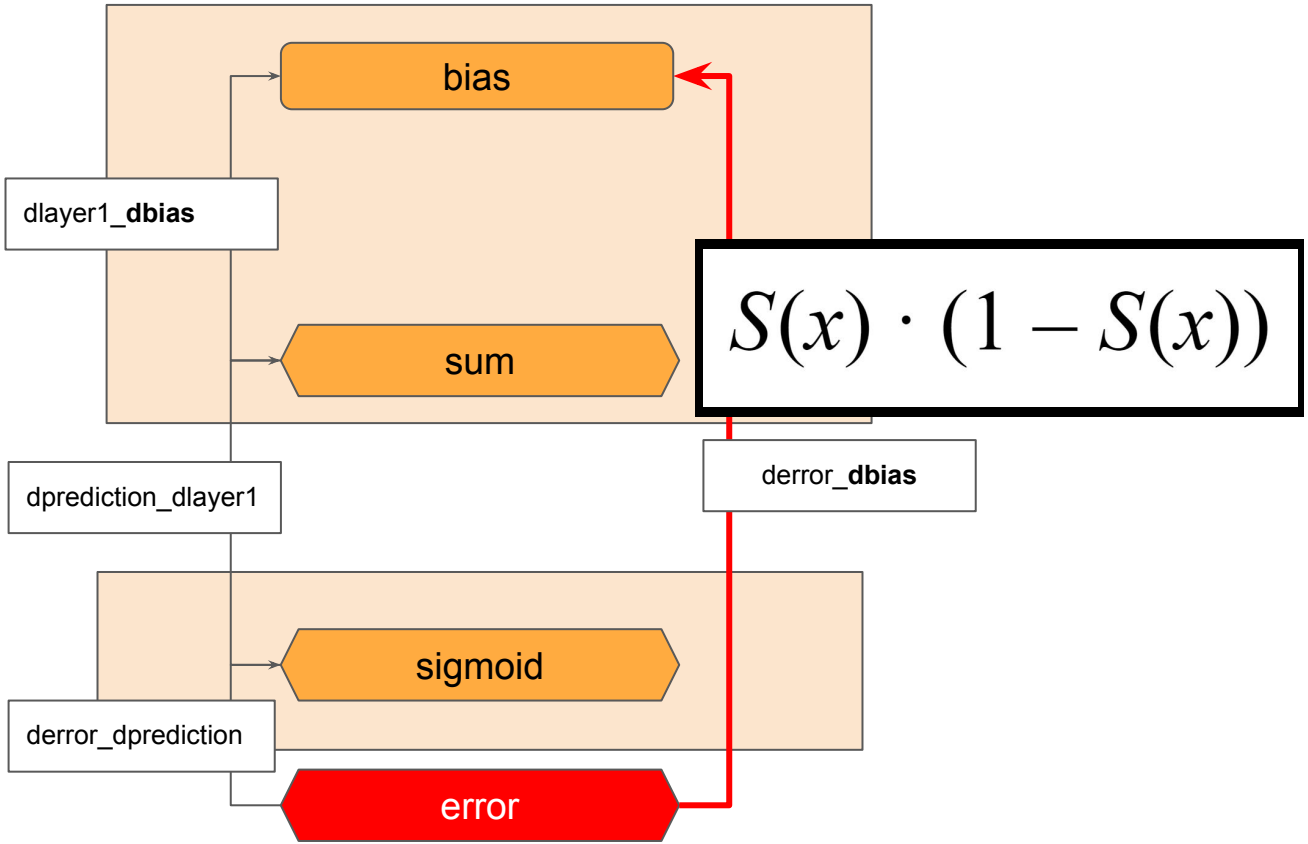
$$
\begin{aligned}
D(x^n) &= n \cdot x^{n-1} \\
D(x^2) &= 2 \cdot x^{2-1} \\
D(x^2) &= 2 \cdot x^1 \\
D(x^2) &= 2x
\end{aligned}
$$

Function Composition

$$error = x^2$$

$$x = prediction - target$$


dot_product(input, **weights**)
dlayer1_**dweights**
sum
dprediction_dlayer1
derror_**dweights**
sigmoid
derror_dprediction
error


bias
dlayer1_dbias
sum
$S(x) \cdot (1 - S(x))$
dprediction_dlayer1
derror_dbias
sigmoid
derror_dprediction
error

```
def sigmoid_deriv(x):
    return sigmoid(x) * (1 - sigmoid(x))

derror_prediction = 2 * (prediction - target)
layer_1 = np.dot(input_vector, weights_1) + bias
dprediction_dlayer1 = sigmoid_deriv(layer_1)
dlayer1_dbias = 1

derror_dbias = (
    derror_prediction * dprediction_dlayer1 * dlayer1_dbias
 )
```

Summary

- Foundations of deep learning and neural networks
- Create a neural network in Python
- Deep learning vs. machine learning
- numpy
- Activation functions
- Backpropagation
- Train a model using a neural network
- TensorFlow, Keras and PyTorch

Resources

- Look Ma, No For-Loops: Array Programming With NumPy
  - https://realpython.com/numpy-array-programming/
- Linear Regression in Python
  - https://realpython.com/linear-regression-in-python/
- Practical Text Classification With Python and Keras
  - https://realpython.com/python-keras-text-classification/
- Pure Python vs NumPy vs TensorFlow Performance Comparison
  - https://realpython.com/numpy-tensorflow-performance/
- PyTorch vs TensorFlow for Your Python Deep Learning Project
  - https://realpython.com/pytorch-vs-tensorflow/

THANK YOU!