AI for Good Workshop

March 2019

Session 3

Artificial Intelligence for everyone

Understands concepts of Deep Learning

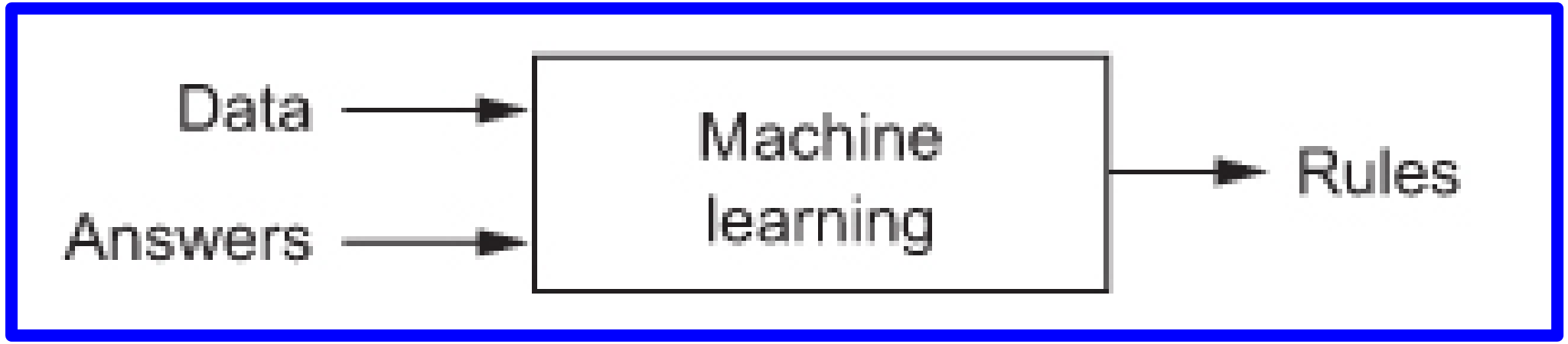
https://sites.google.com/view/AIforEveryone

+

Topics in this presentation

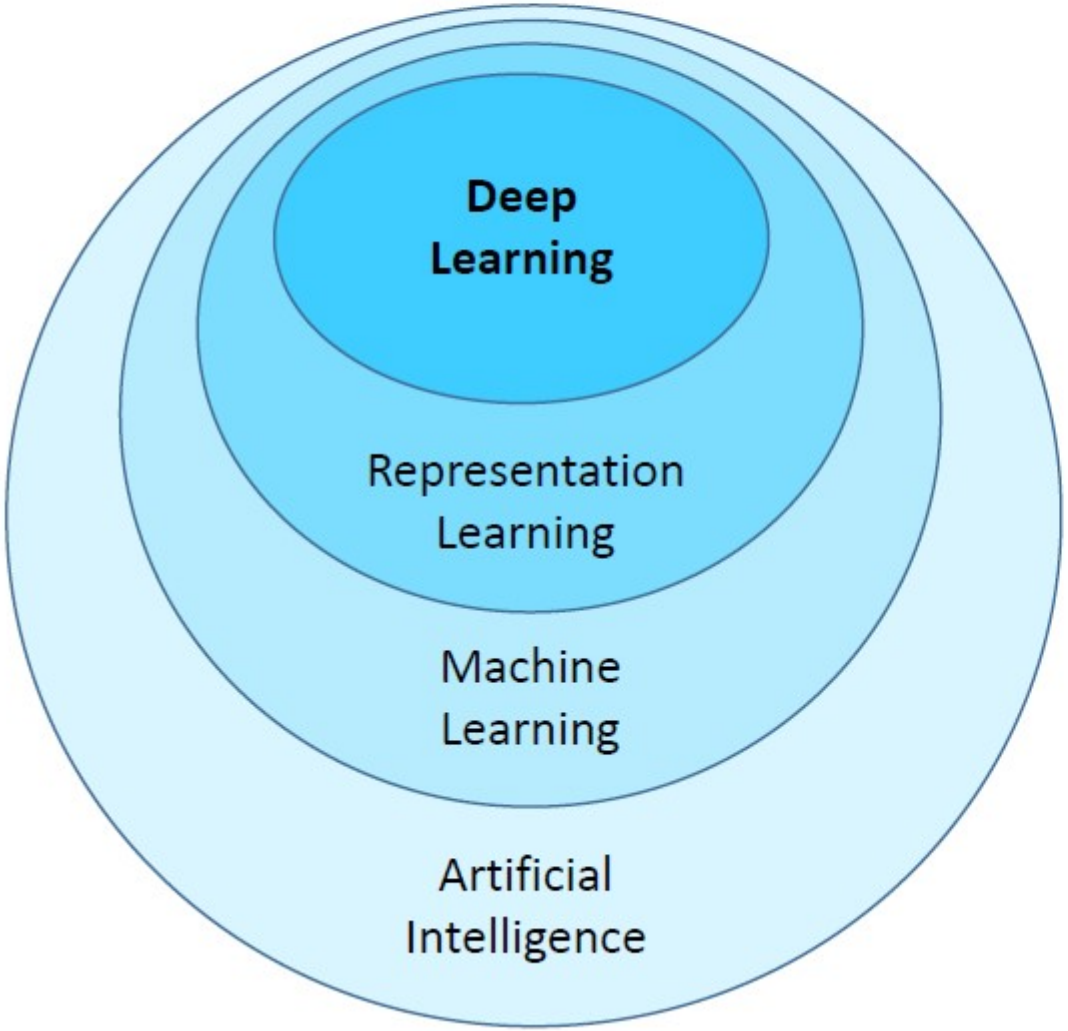
Basic conceptual understanding of Neural Networks, Gradient Descent

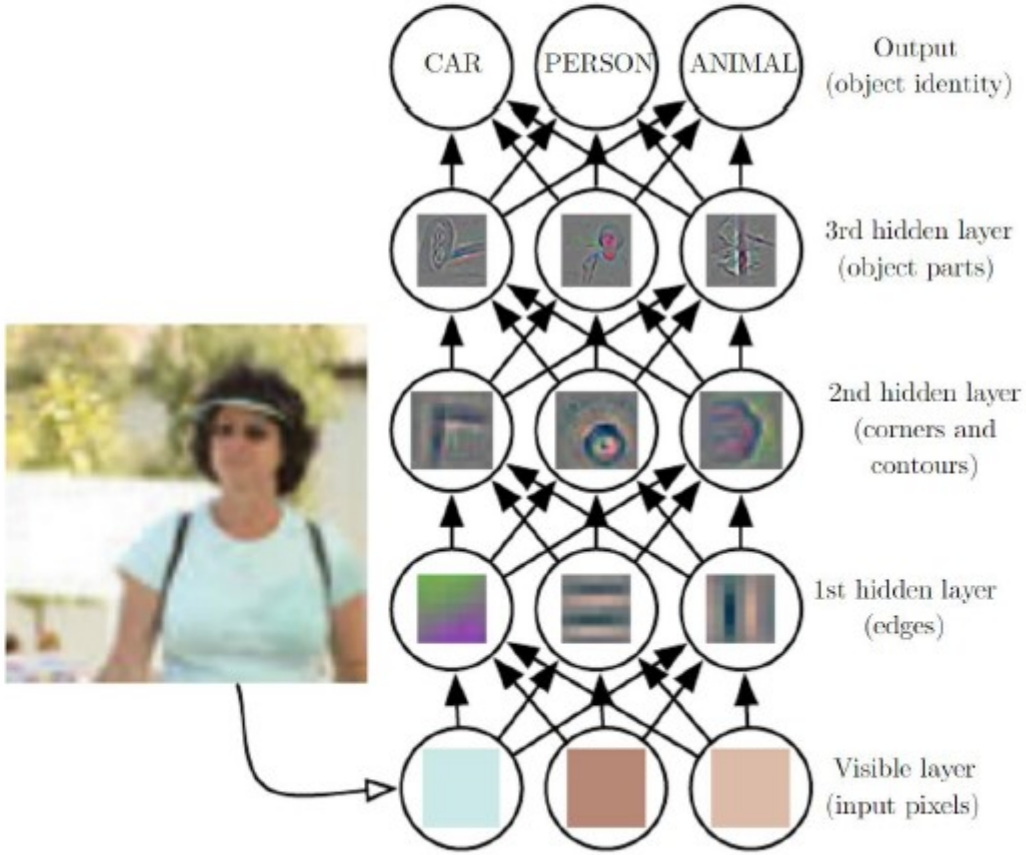
New way to think from today !



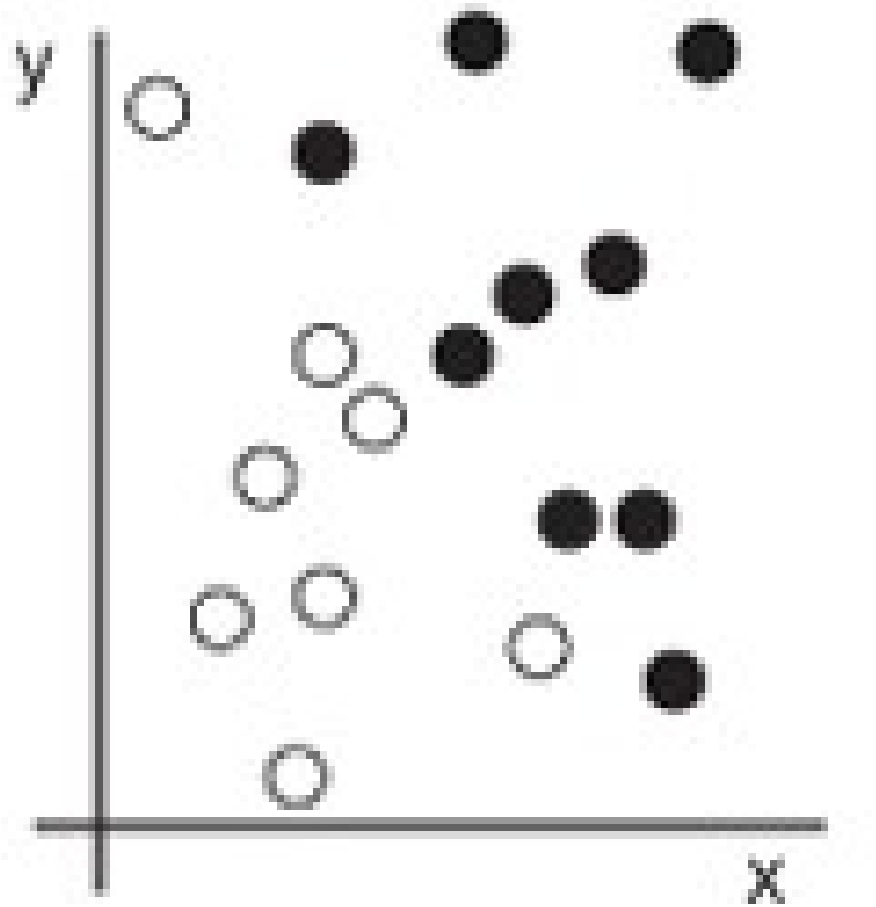
Deep Learning is **Representation Learning**

(aka Feature Learning)

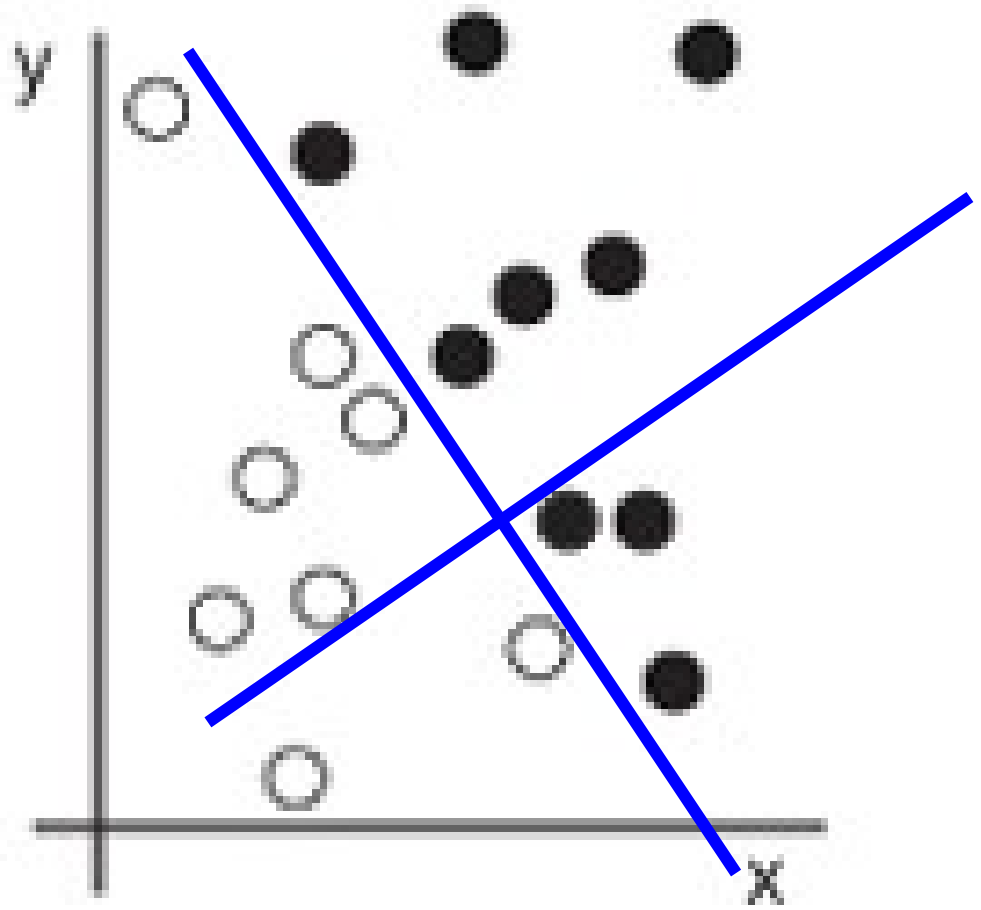




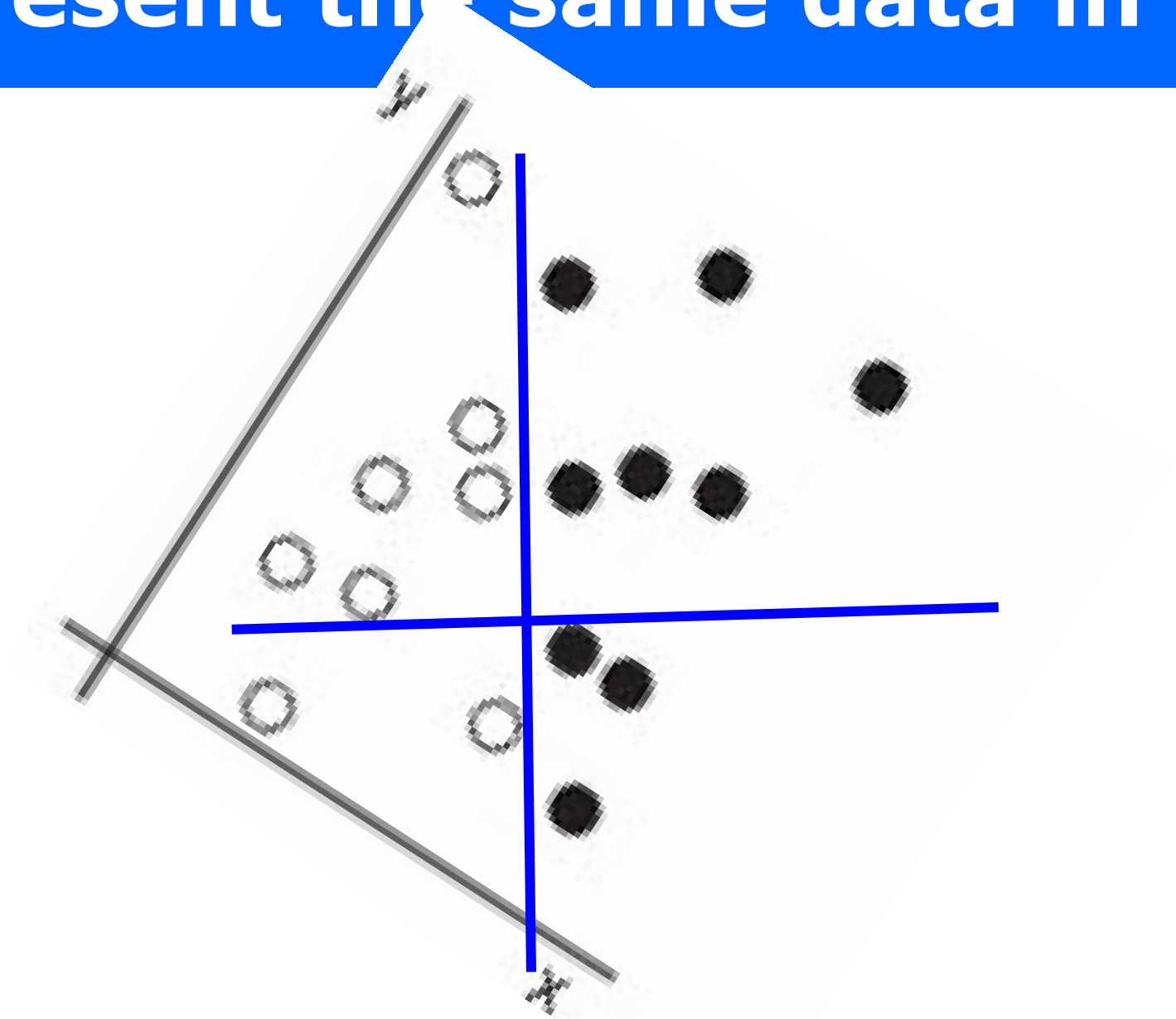
Represent the same data in a new way



Represent the same data in a new way



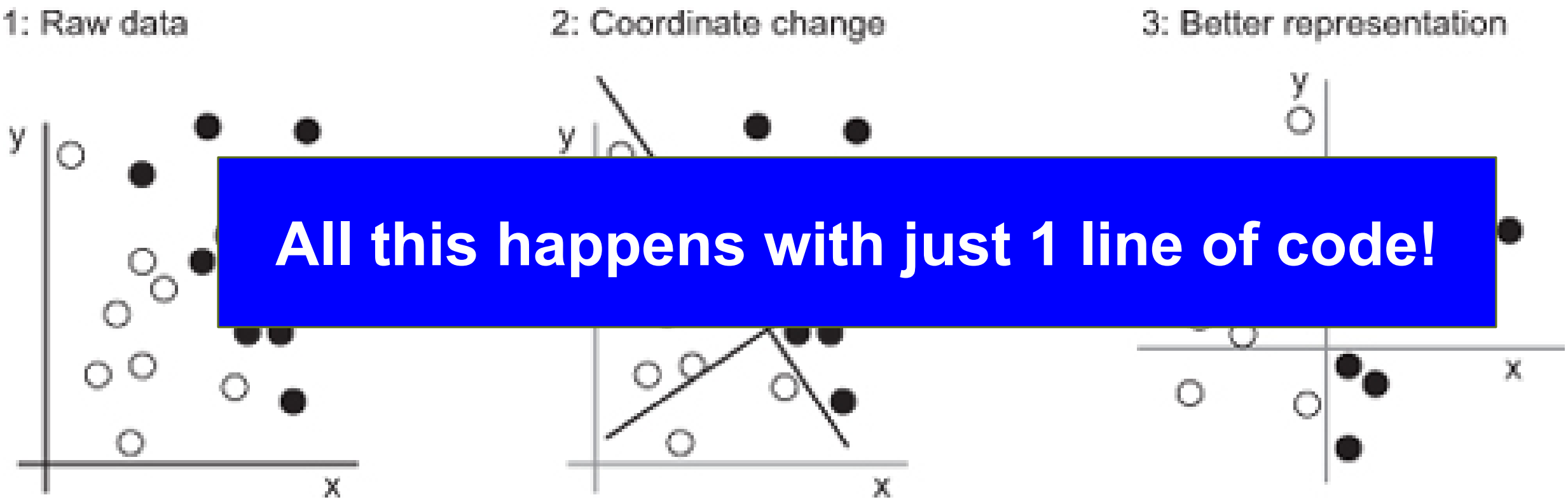
Represent the same data in a new way

- 

Find a way to represent the data in a better way?

Deep learning is === a multistage way to learn data representations.

It's a simple idea—but, as it turns out, very simple mechanisms, sufficiently scaled, can end up looking like magic.



All this happens with just 1 line of code!

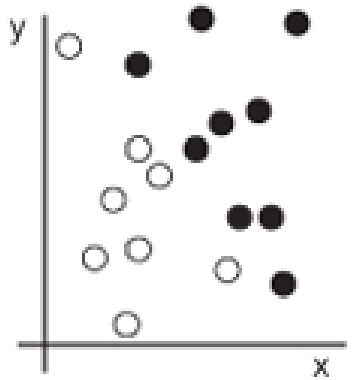
Just 1 line of code == magic!

Deep learning is === a multistage way to learn data representations.

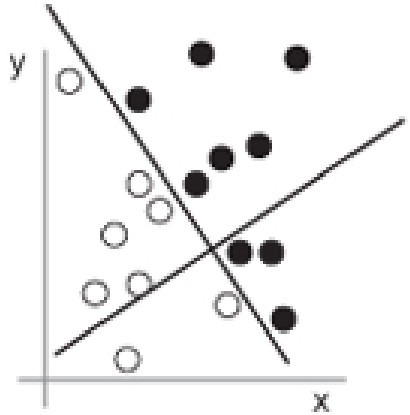
It's a simple idea—but, as it turns out, very simple mechanisms, sufficiently scaled, can end up looking like magic.

myModel. fit (X , Y)

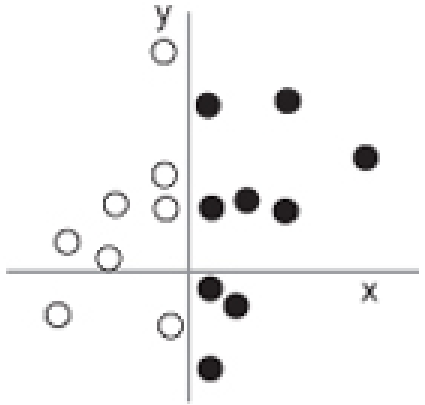
1: Raw data



2: Coordinate change



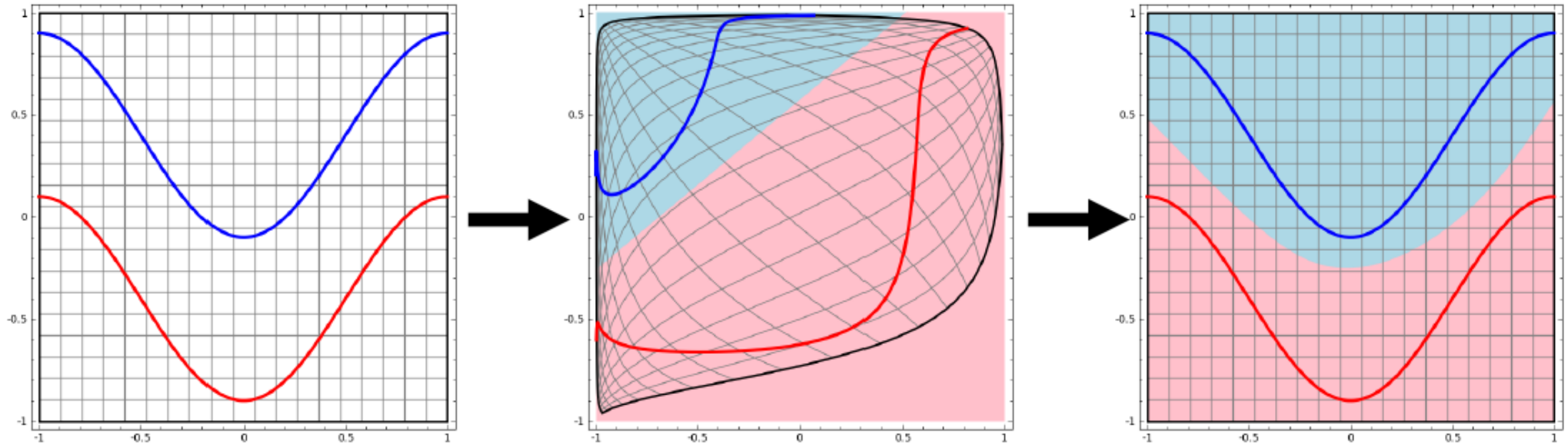
3: Better representation



Deep Learning is **Representation Learning**

(aka Feature Learning)

All this happens with just 1 line of code!

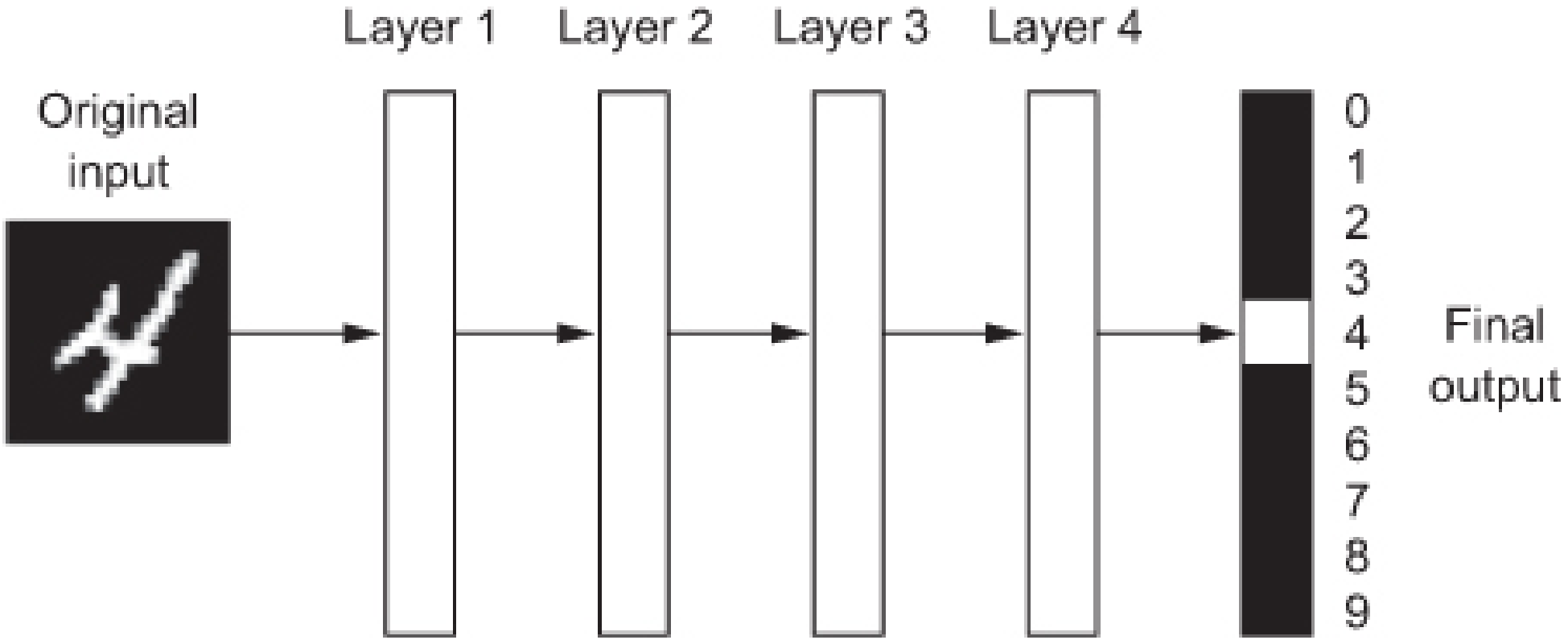


Task: Draw a line to separate the **blue curve** and **red curve**

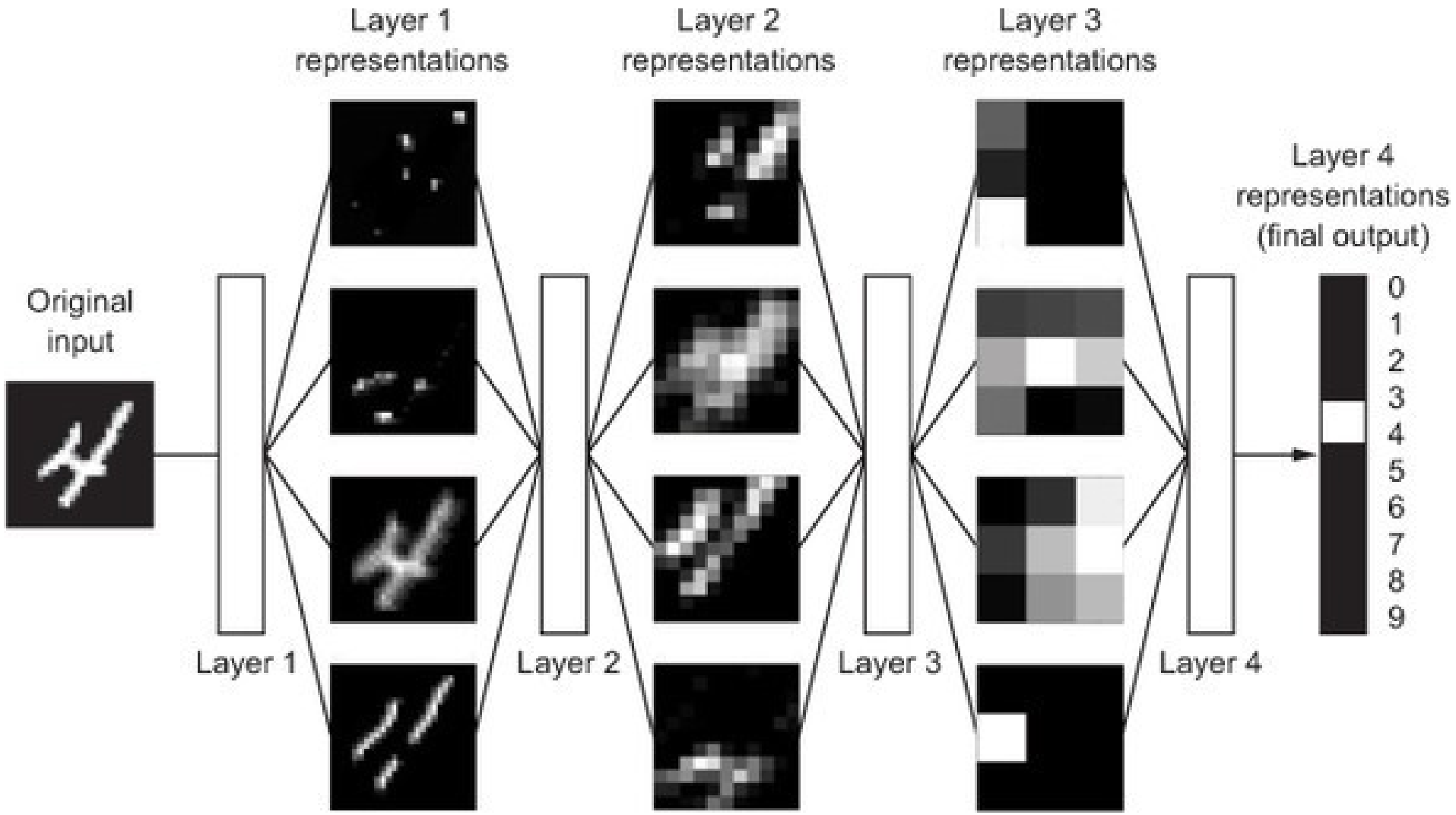
How did the magic happen?

```
import keras

(x, y) = load_data()

model = keras.Sequential()

model.add(layers.Dense)
model.add(layers.Dense)

model.fit(x, y)
```

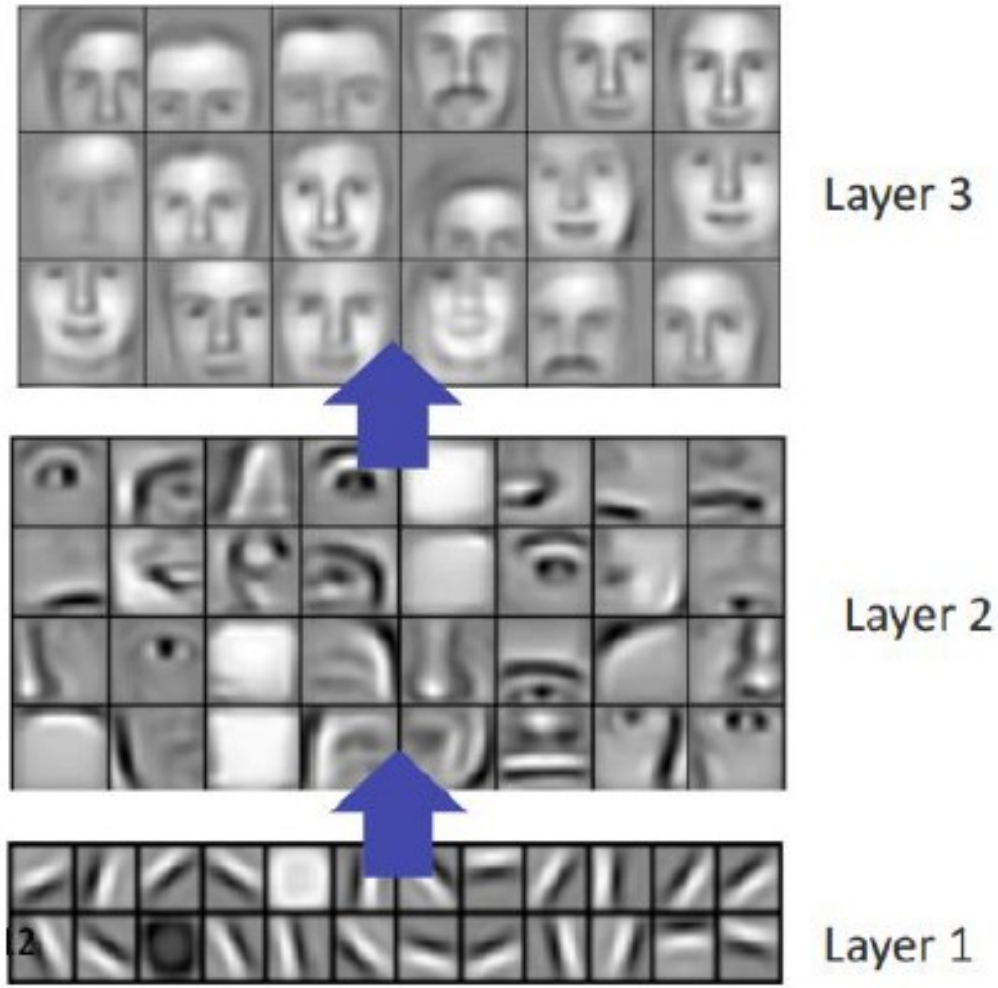
Can a neural network automatically represent the raw data into another way?



Deep representations learned by a neural network model

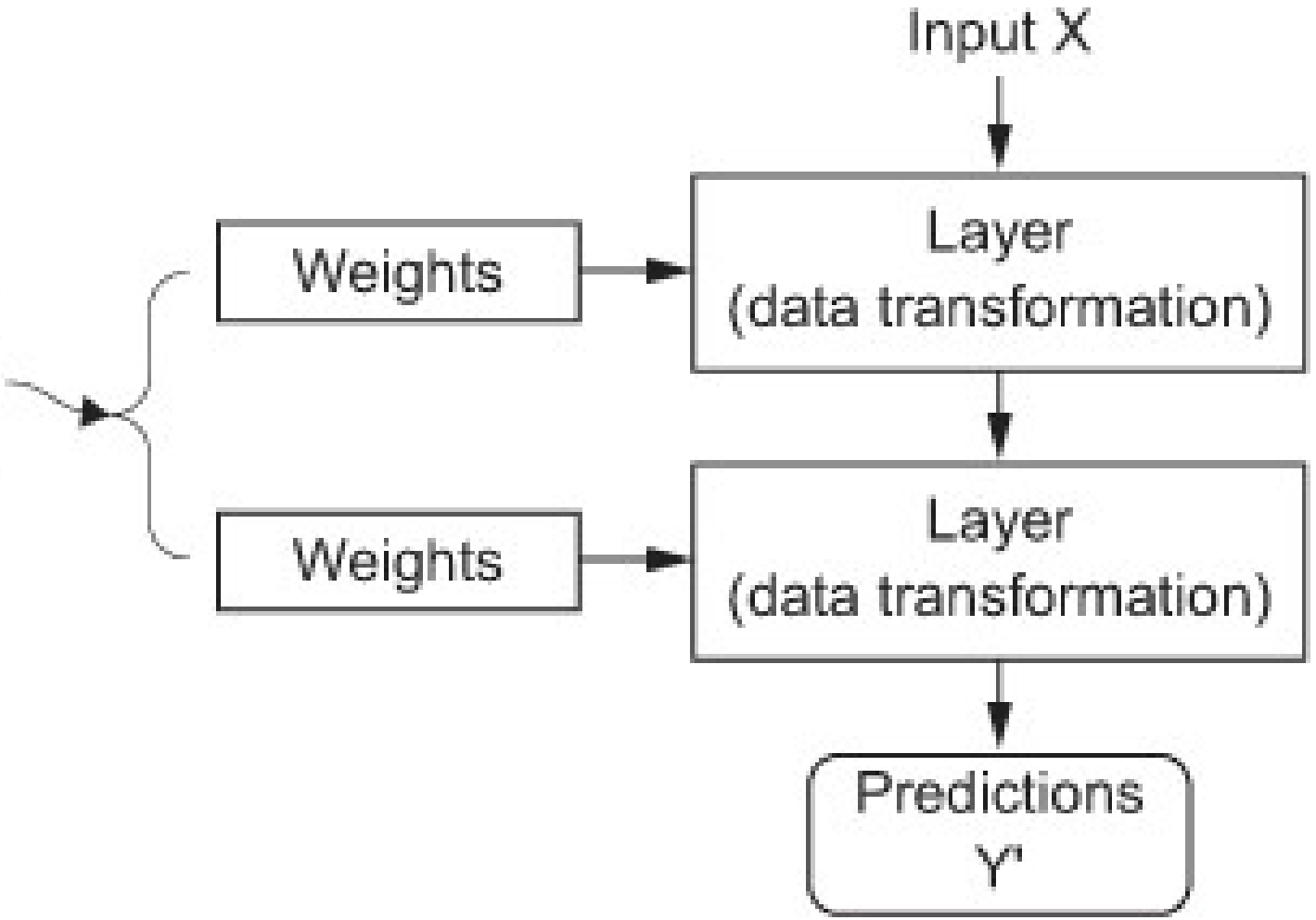


representations “learned” automatically

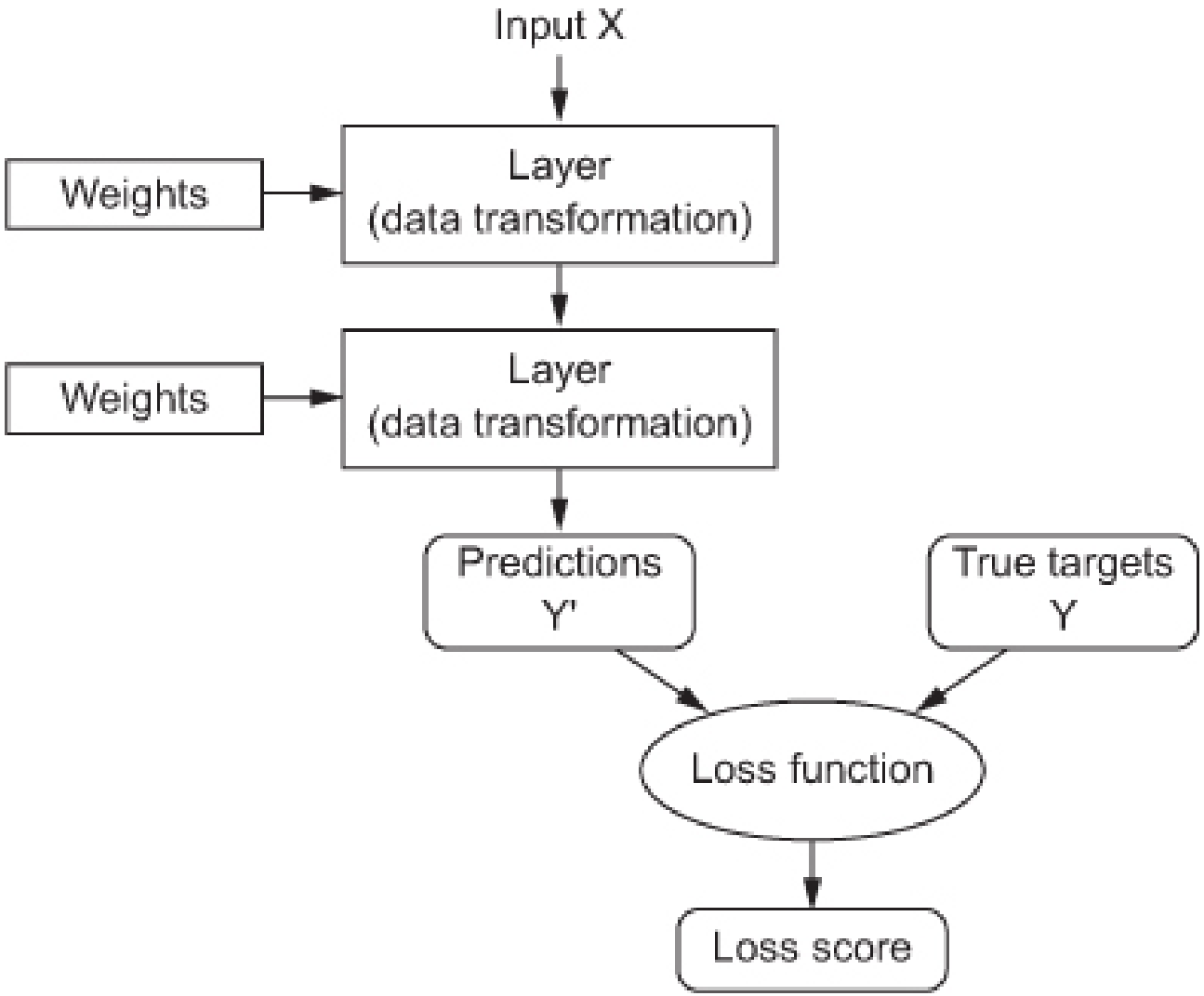


A neural network is parameterized by its weights

Goal: finding the right values for these weights



A loss function measures the quality of the network's output



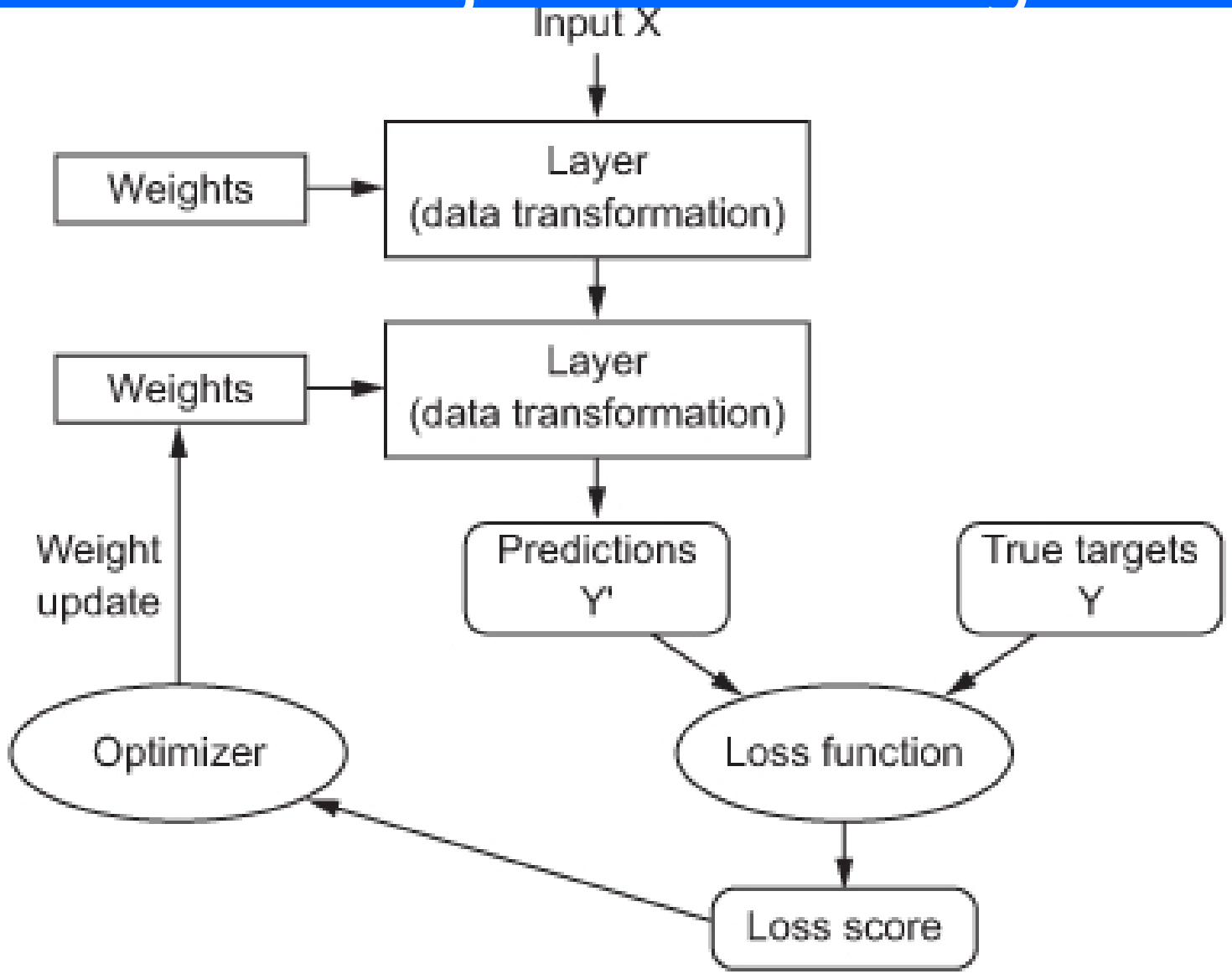
The loss function takes the predictions of the network and the true target.

and computes a distance score,

capturing how well the network has done on this specific example

Credits, Deep Learning with Python

The loss score is used as a feedback signal to adjust the weights

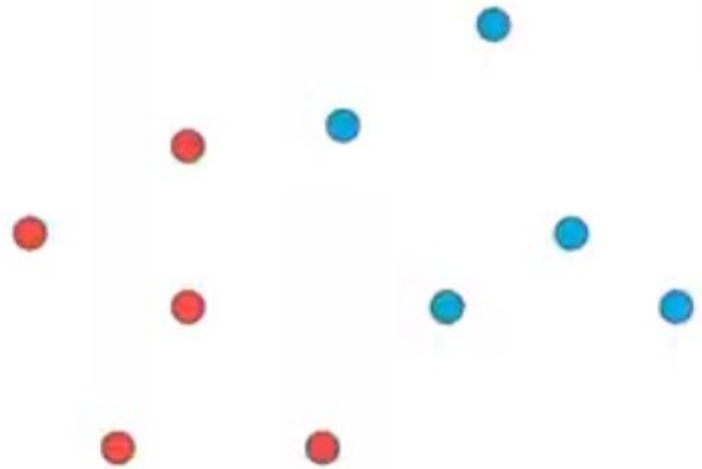


Neural Networks

Credits: Luis Serrano

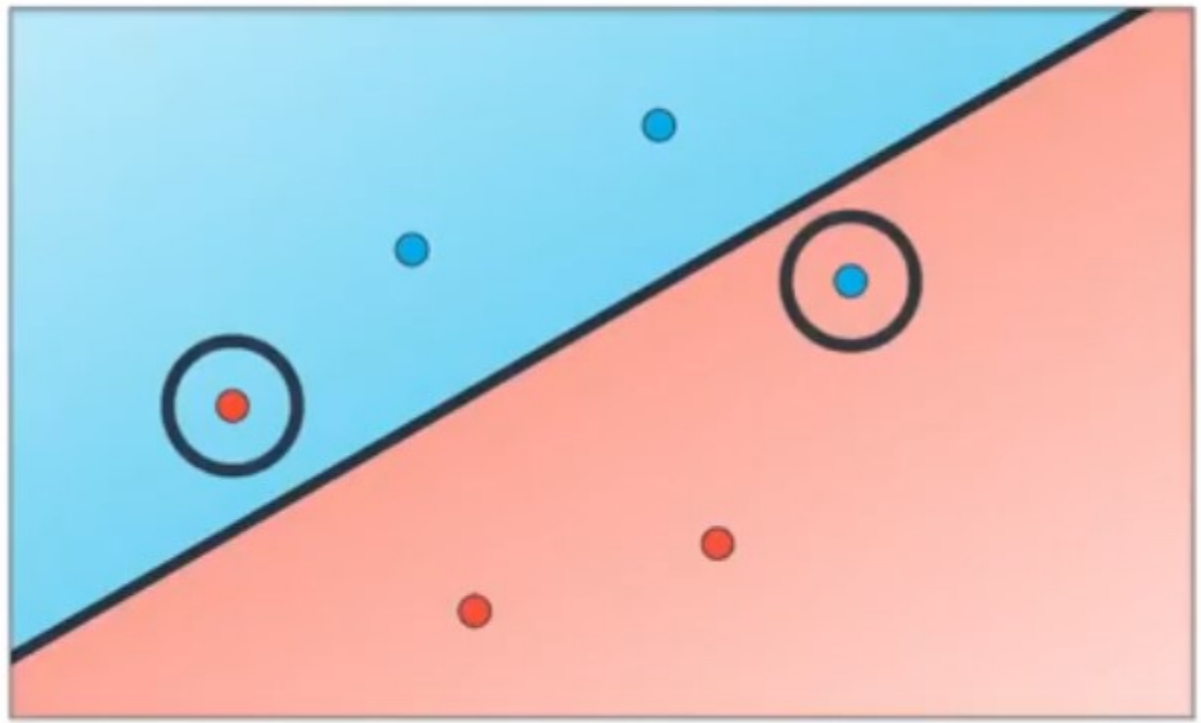
https://www.youtube.com/watch?v=BR9h47Jtqyw

Goal: Split Data

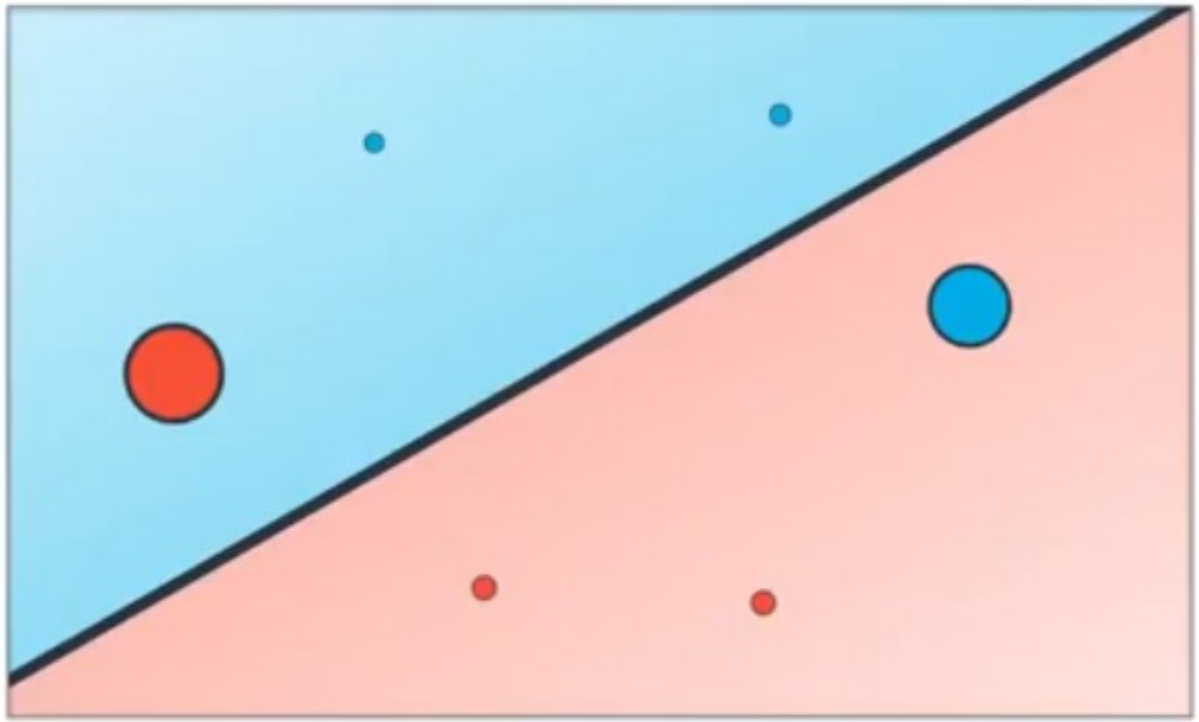


Teach a computer how to split the data

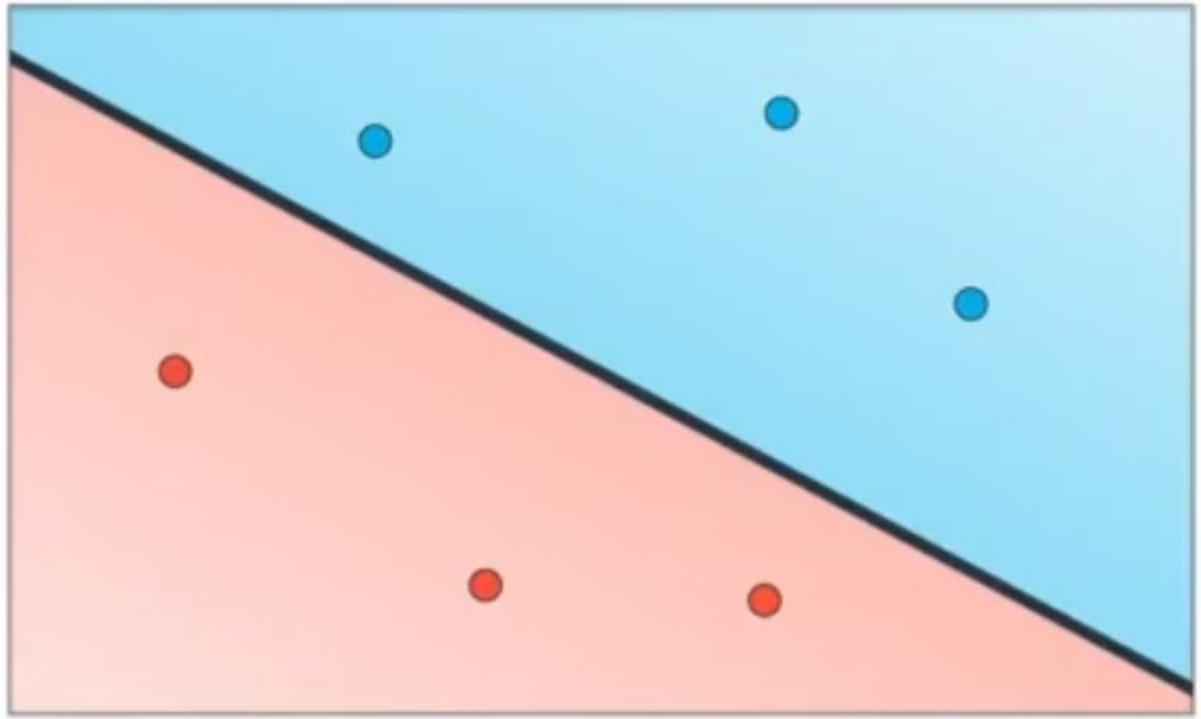
Goal: Split Data



2 errors



Error = • + • + ● + ● + ● + ●



Error = • + • + ● + ● + • + •

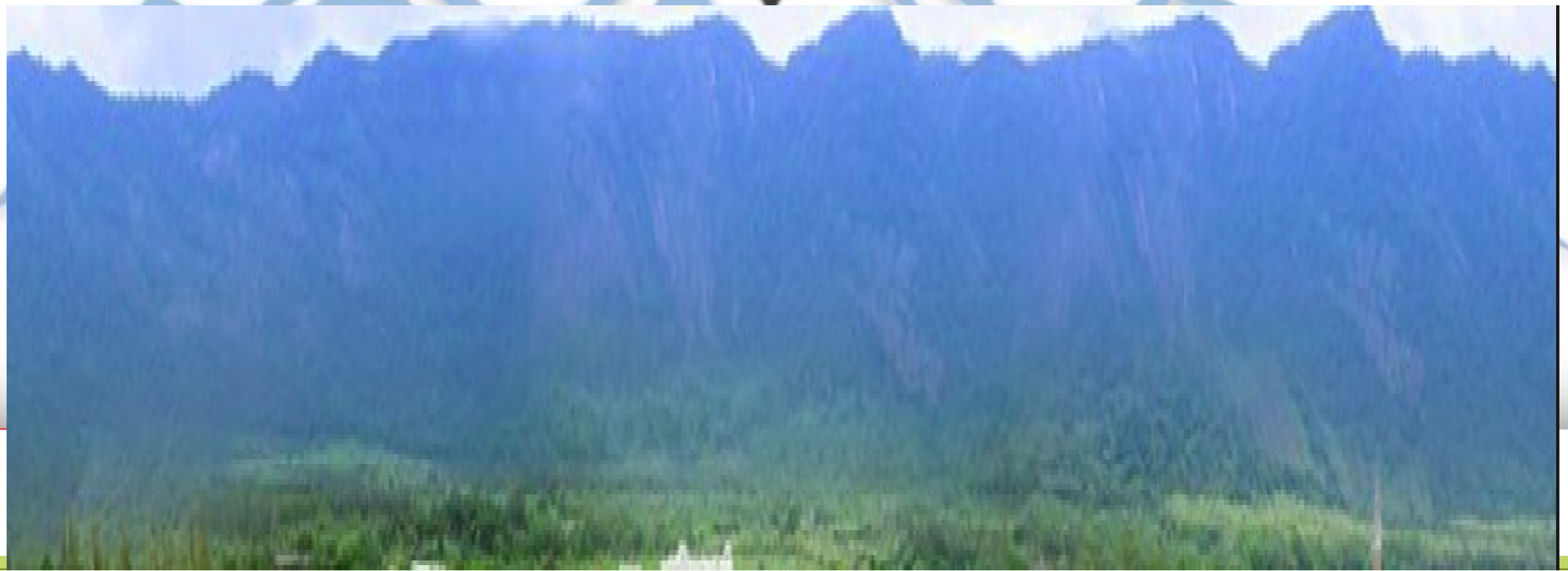
Error = ● + ● + ● + ● + ● + ●

Minimize error

Gradient descent

Group Activity game to learn Gradient Descent

Gradient descent

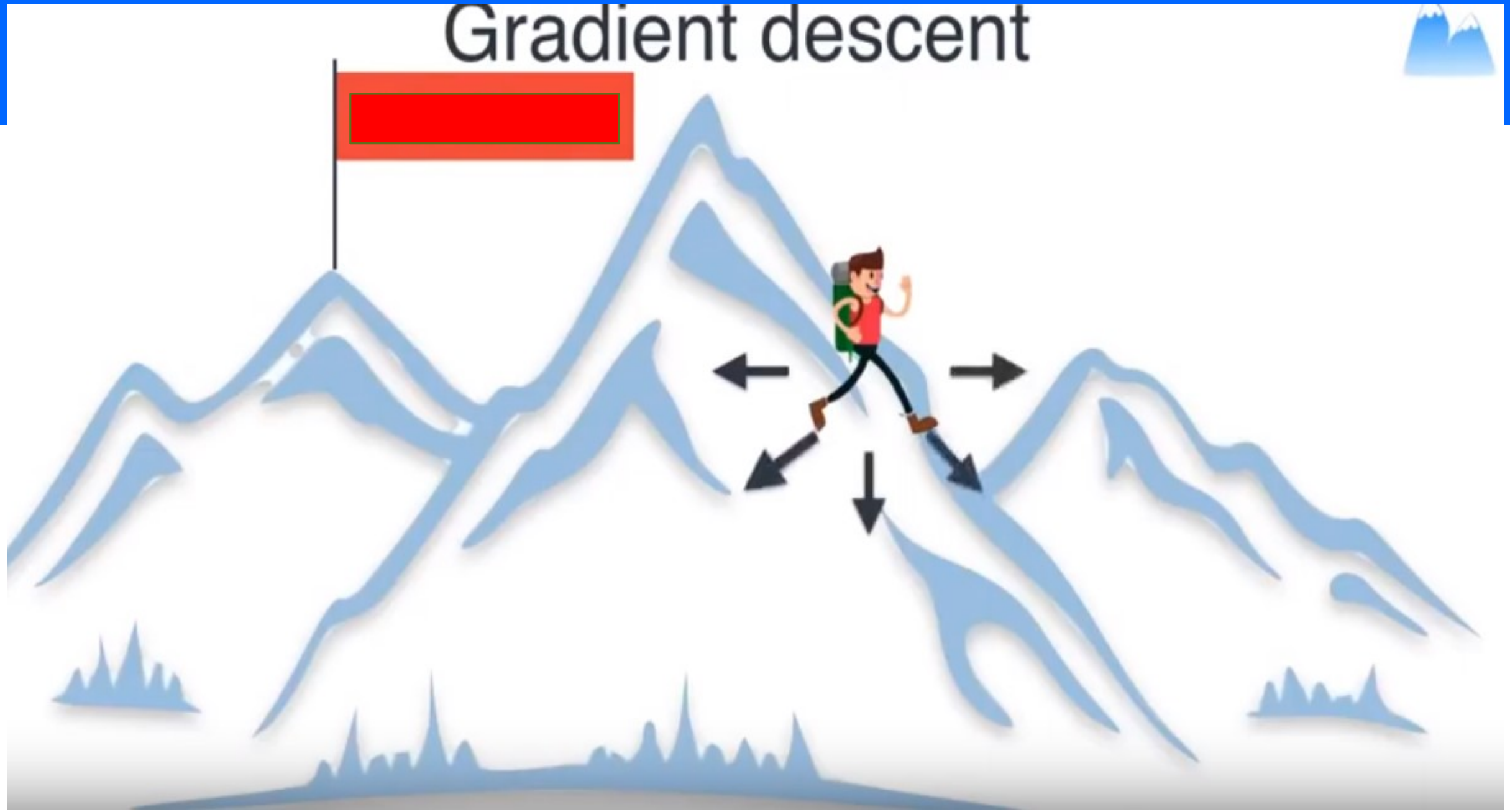


Gradient descent

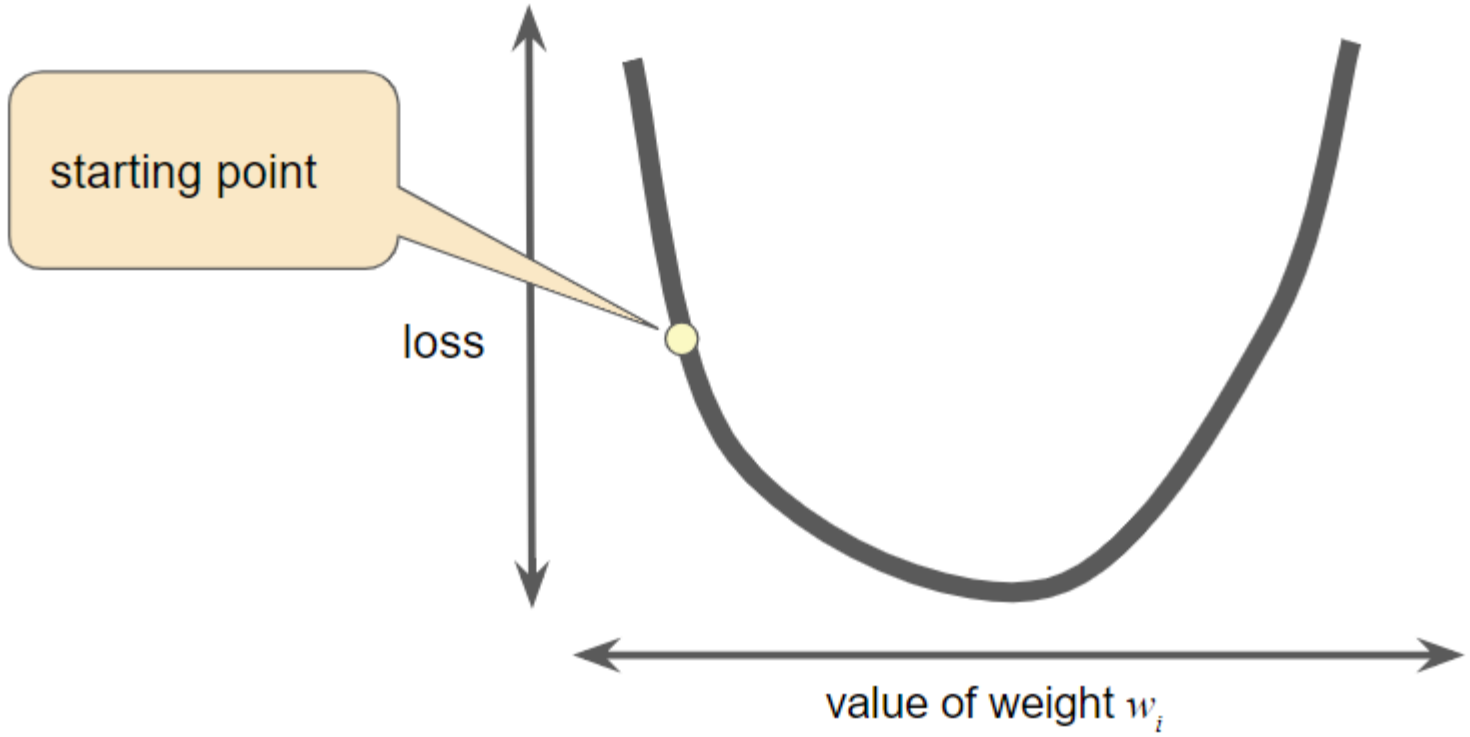
Credits: Luis Serrano

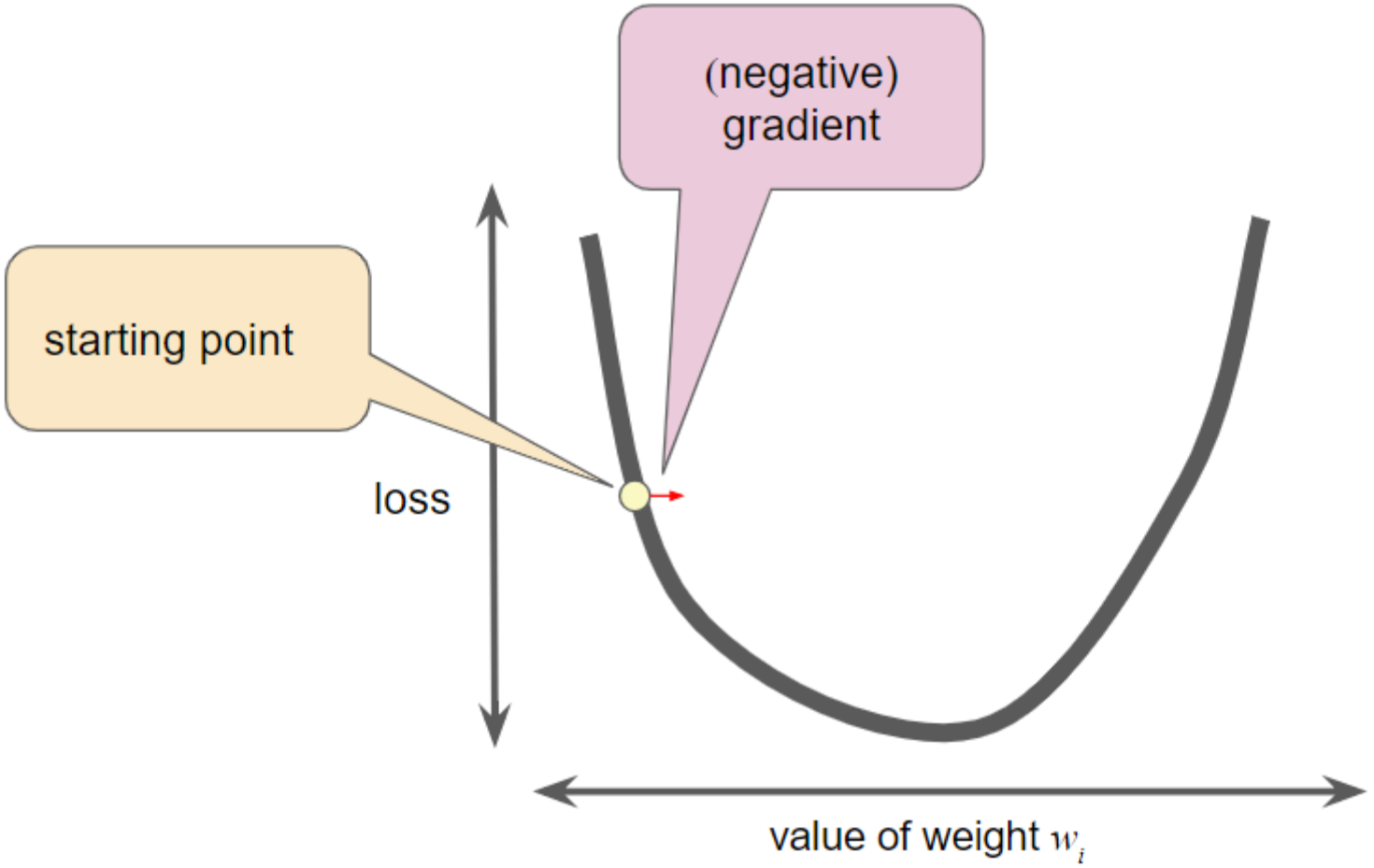
https://www.youtube.com/watch?v=BR9h47Jtqyw

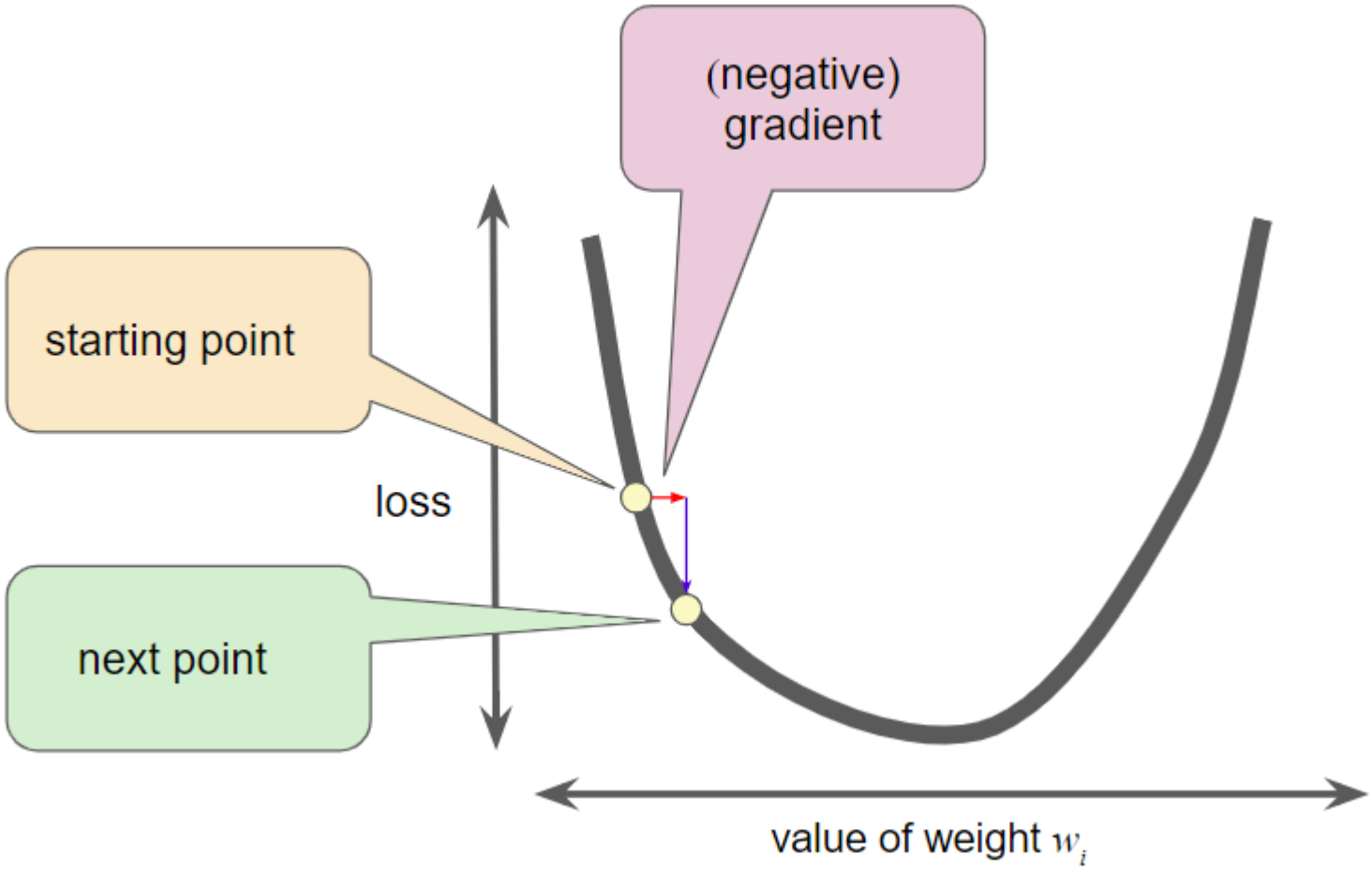
Gradient descent



Gradient descent

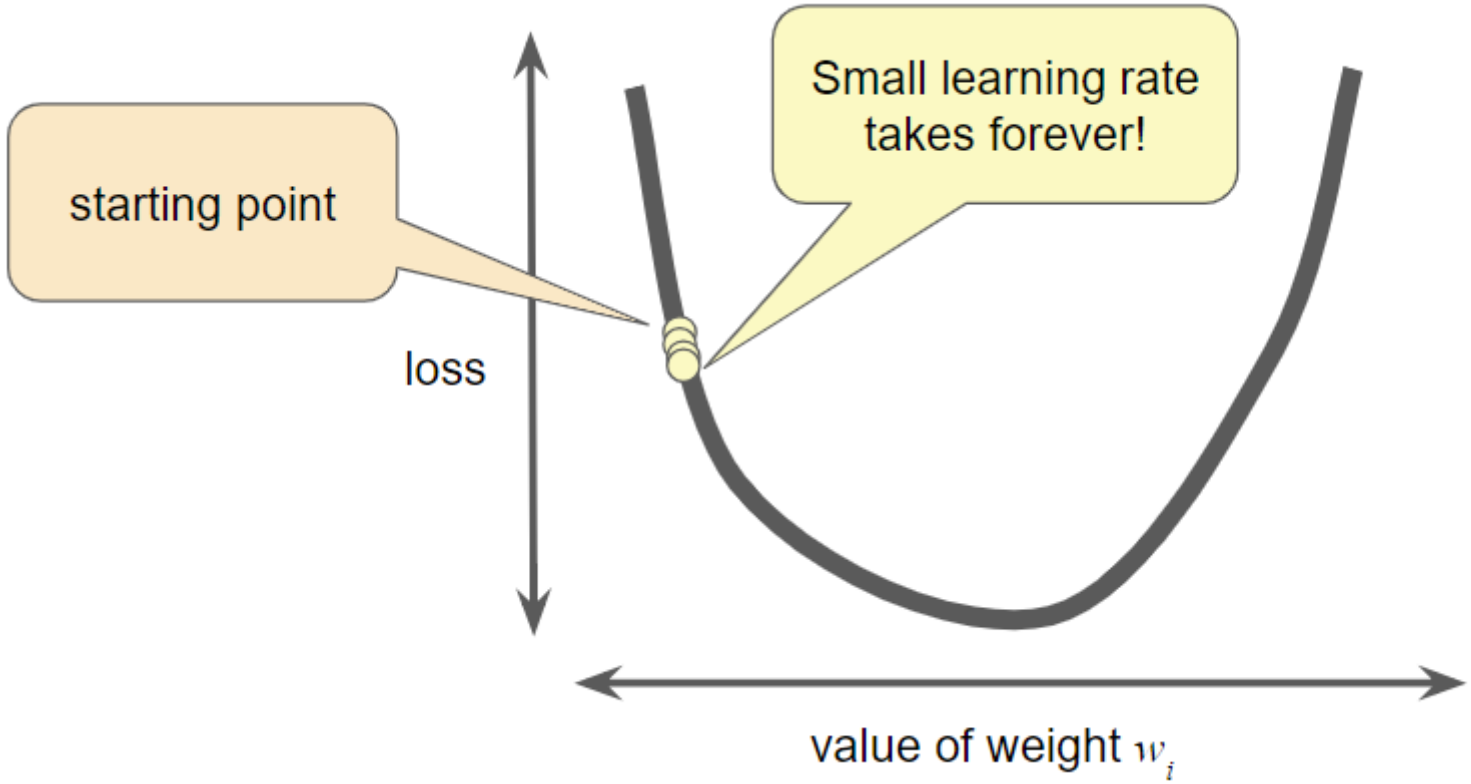


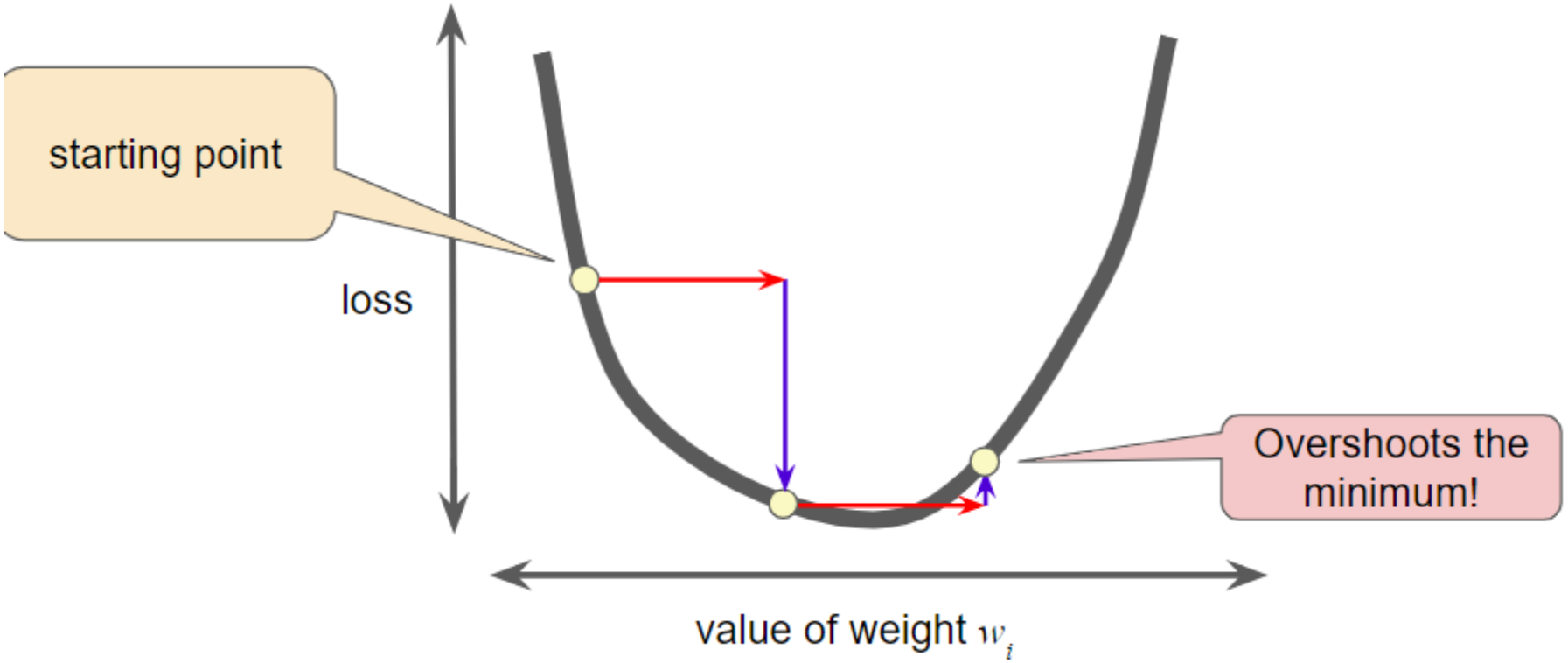

(negative)
gradient
starting point
loss
value of weight w_i

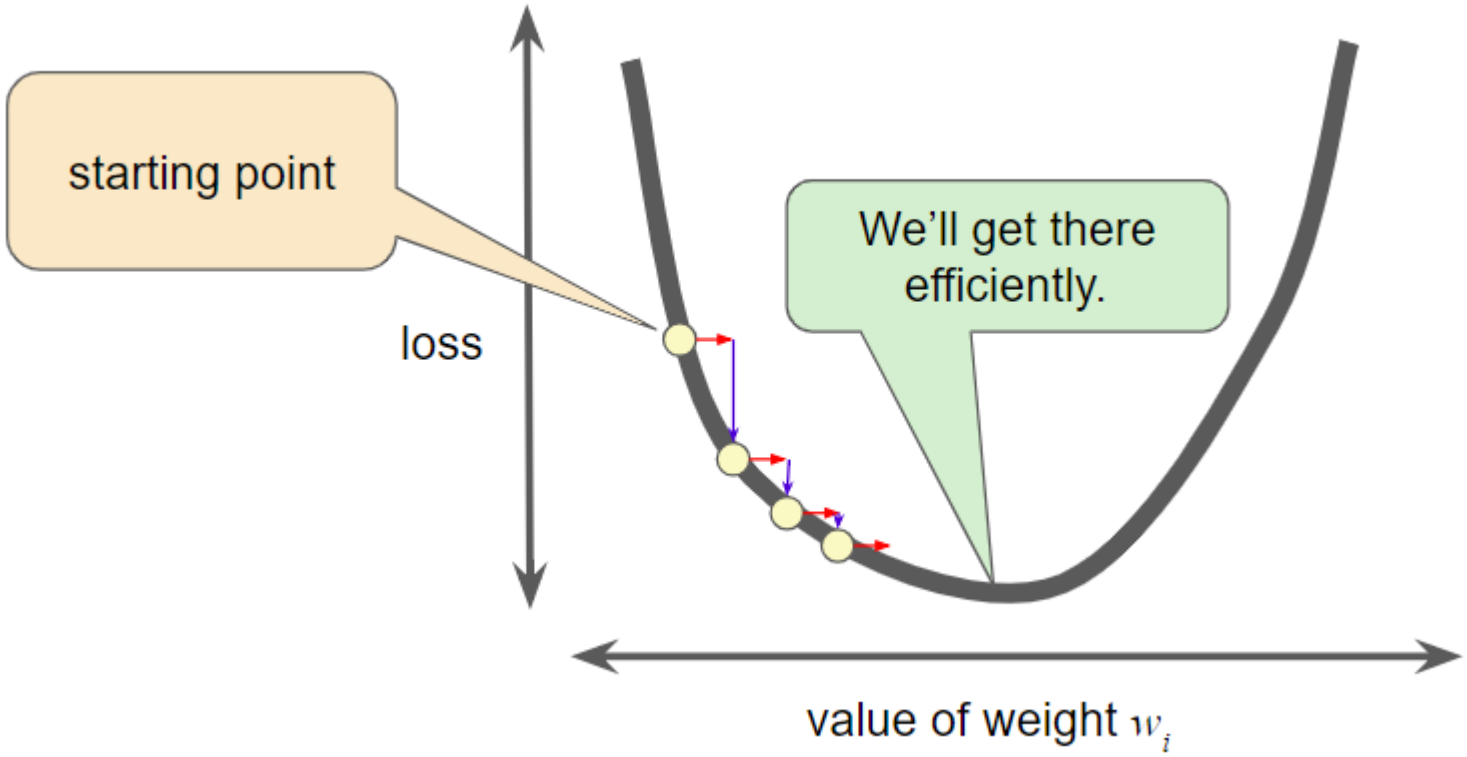


Credits: Google Machine Learning Crash Course

https://developers.google.com/machine-learning/crash-course/

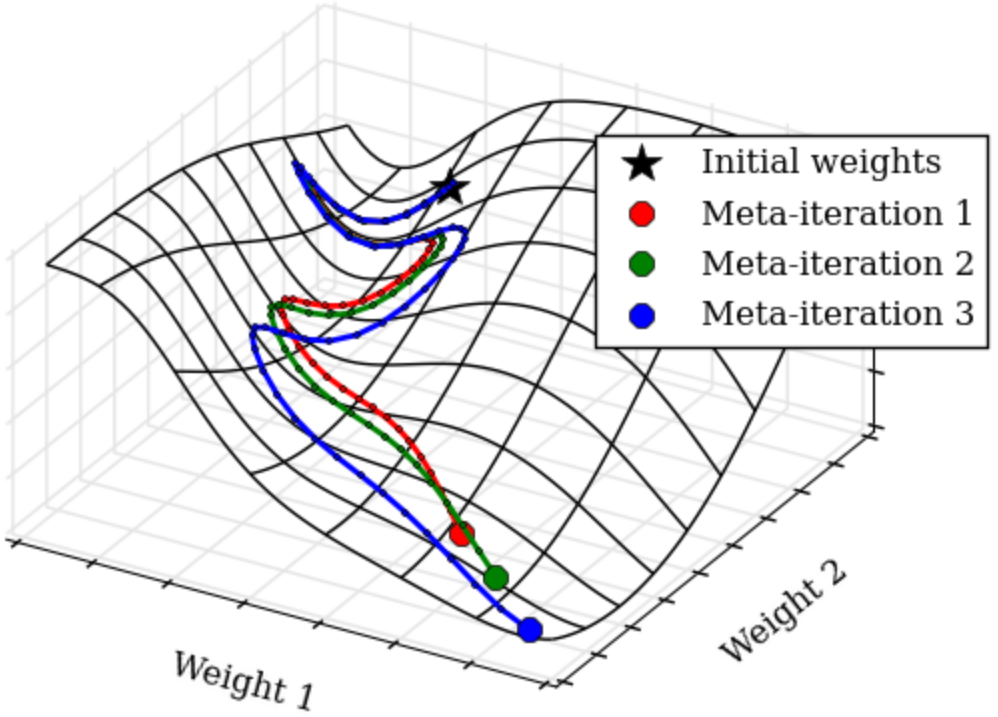

Small learning rate takes forever!
starting point
loss
value of weight w_i

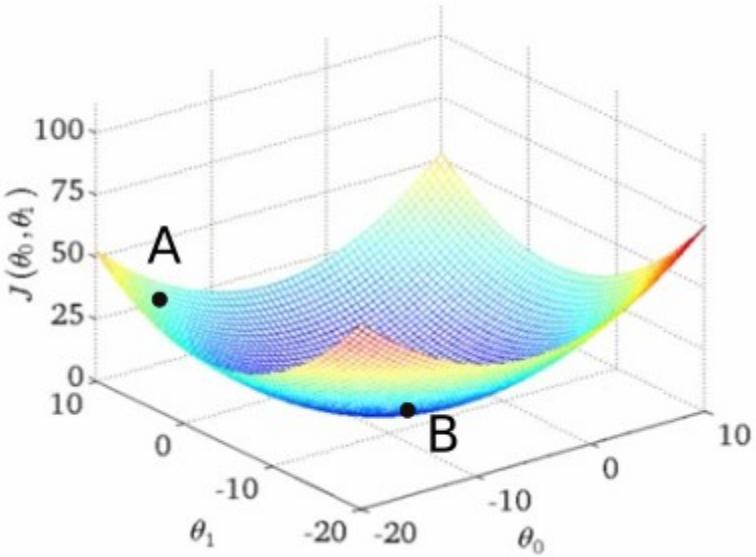

starting point
loss
Overshoots the minimum!
value of weight w_i

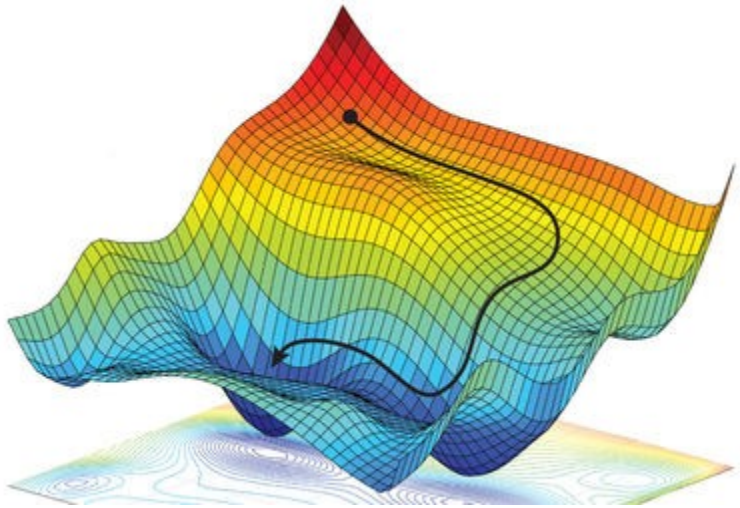


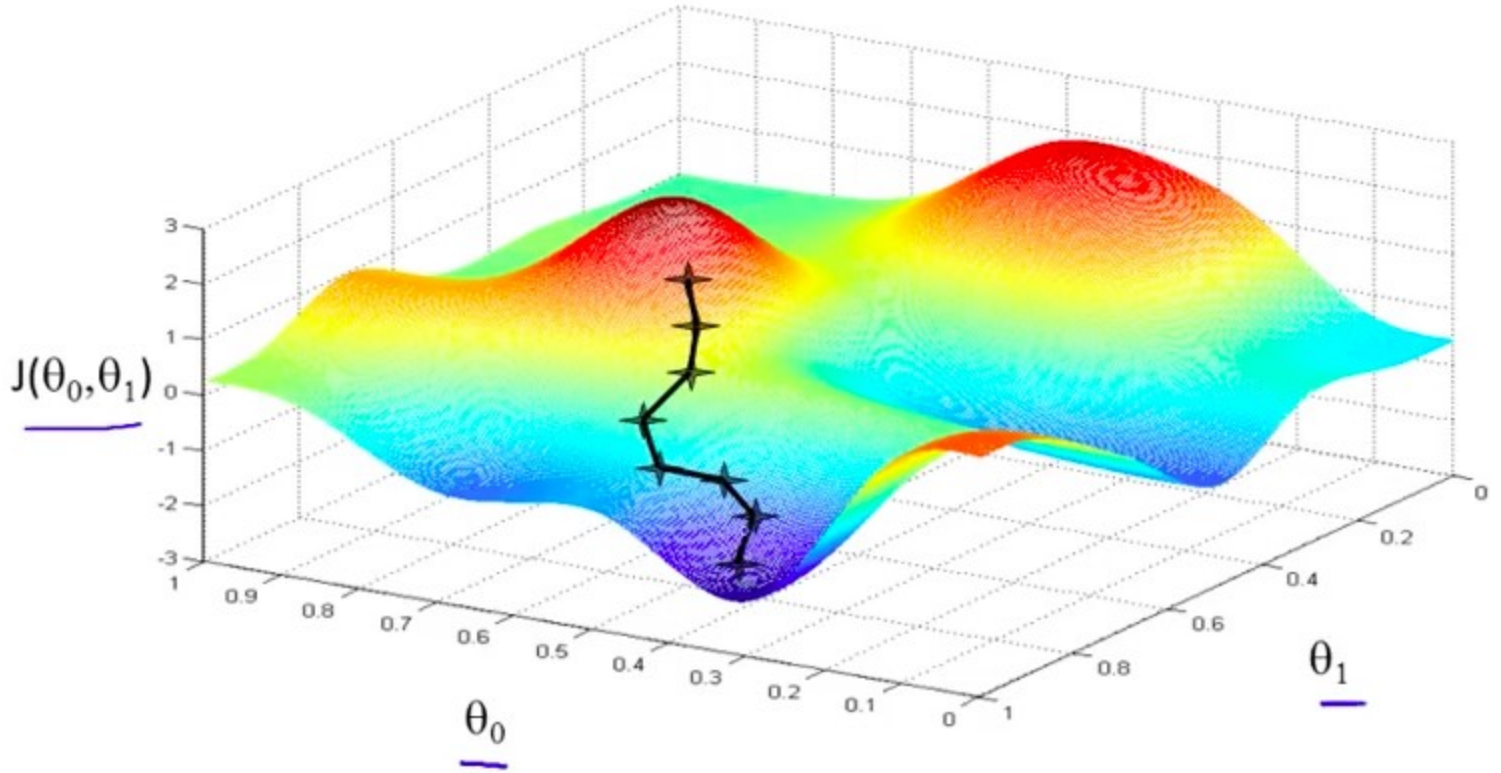
Credits: Google Machine Learning Crash Course

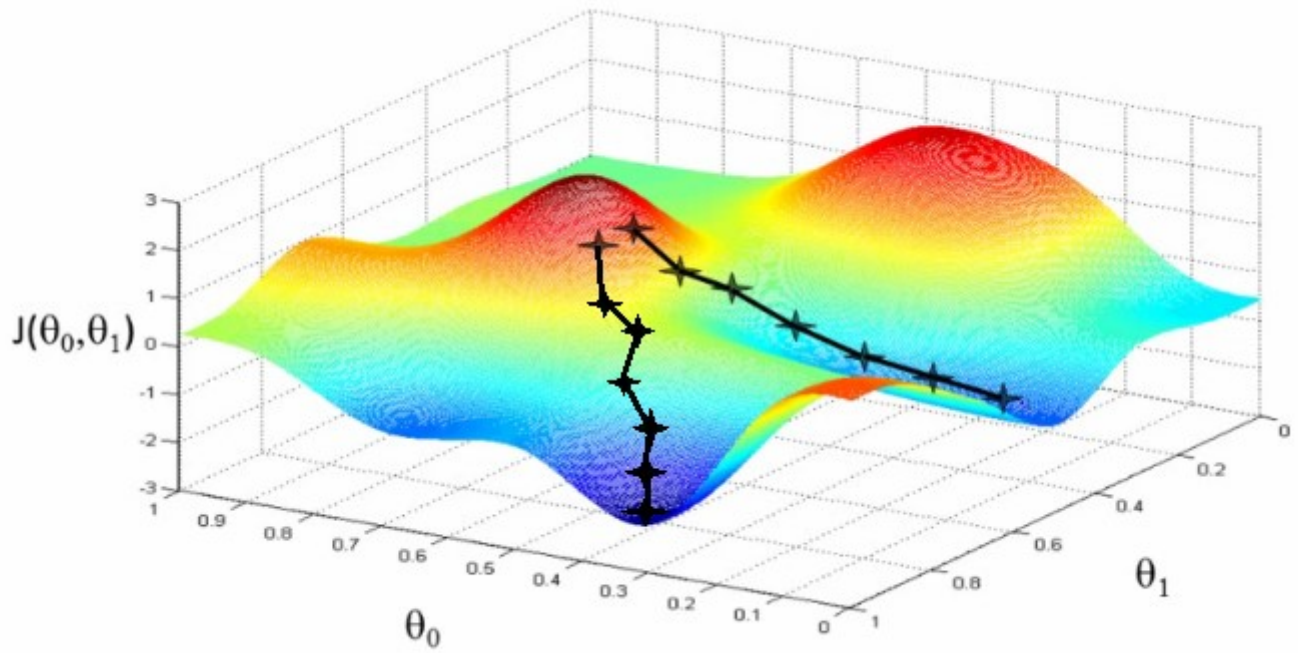
https://developers.google.com/machine-learning/crash-course/


Initial weights
Meta-iteration 1
Meta-iteration 2
Meta-iteration 3
Weight 1
Weight 2


100
75
50
25
0
$J(\theta_0, \theta_1)$
A
B
10
0
-10
-20
θ_1
-20
-10
0
10
θ_0

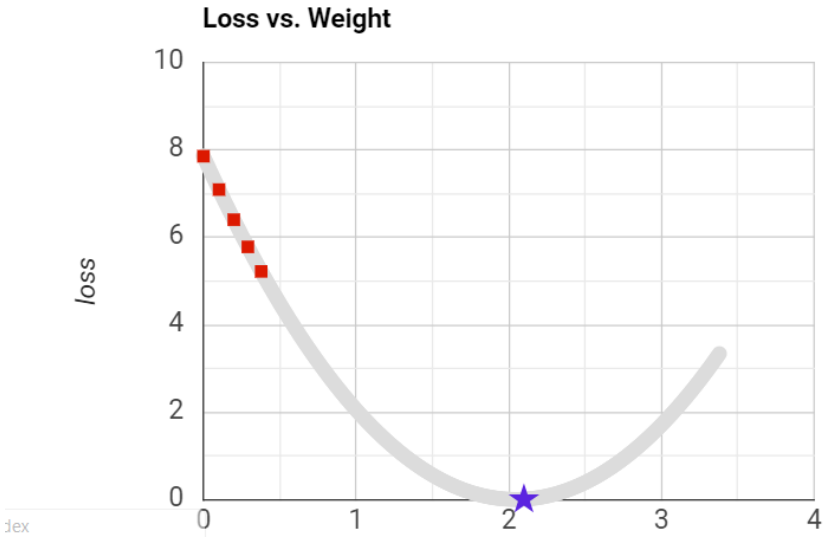




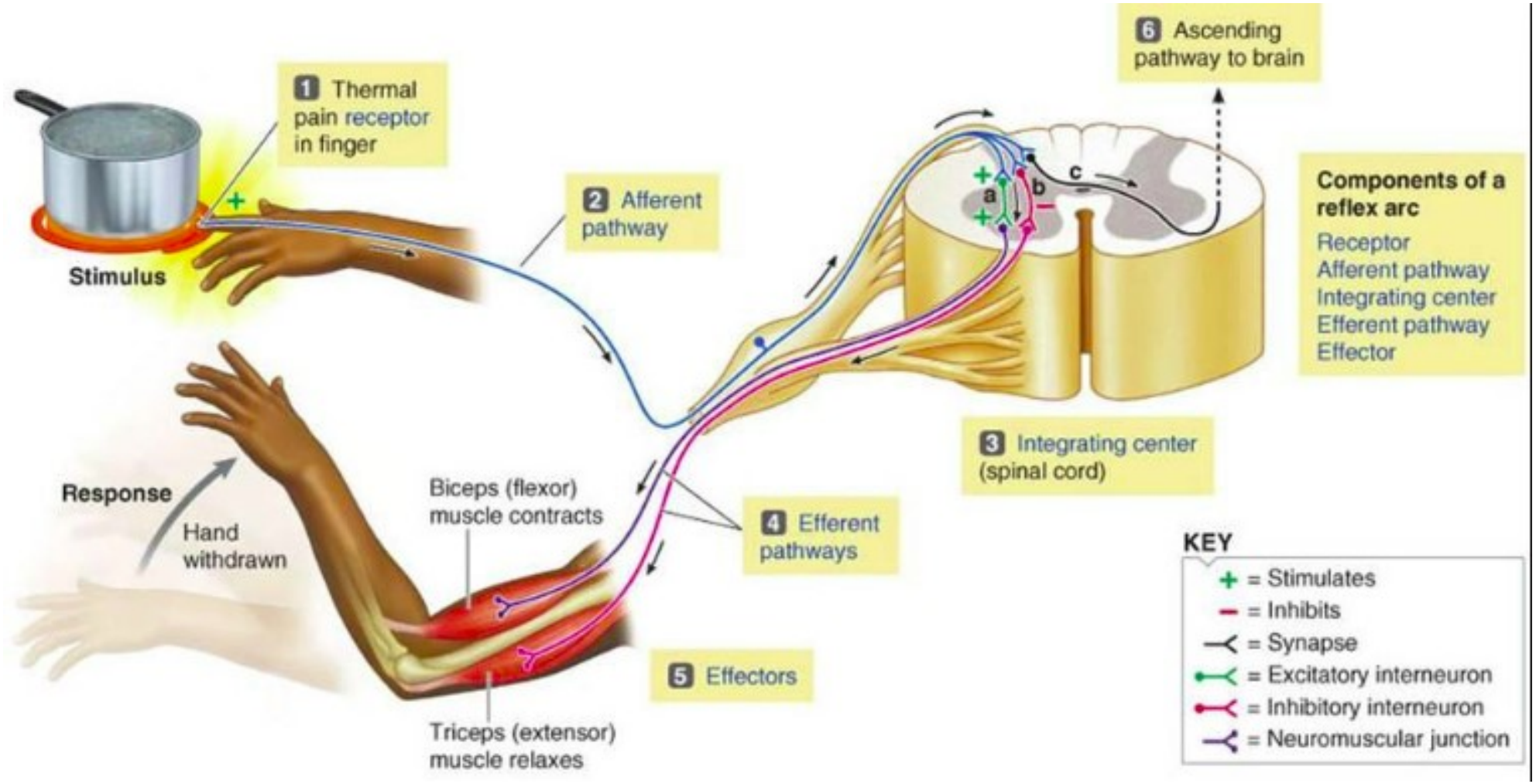


What should be Learning rate ?

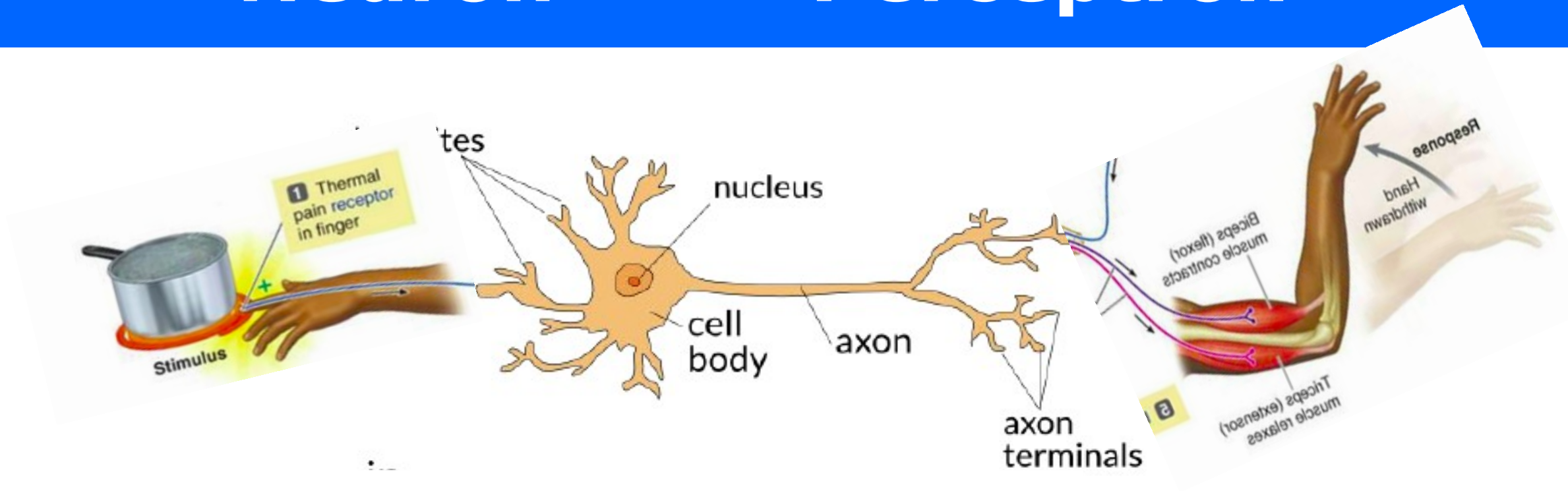
- https://developers.google.com/machine-learning/crash-course/fitter/graph



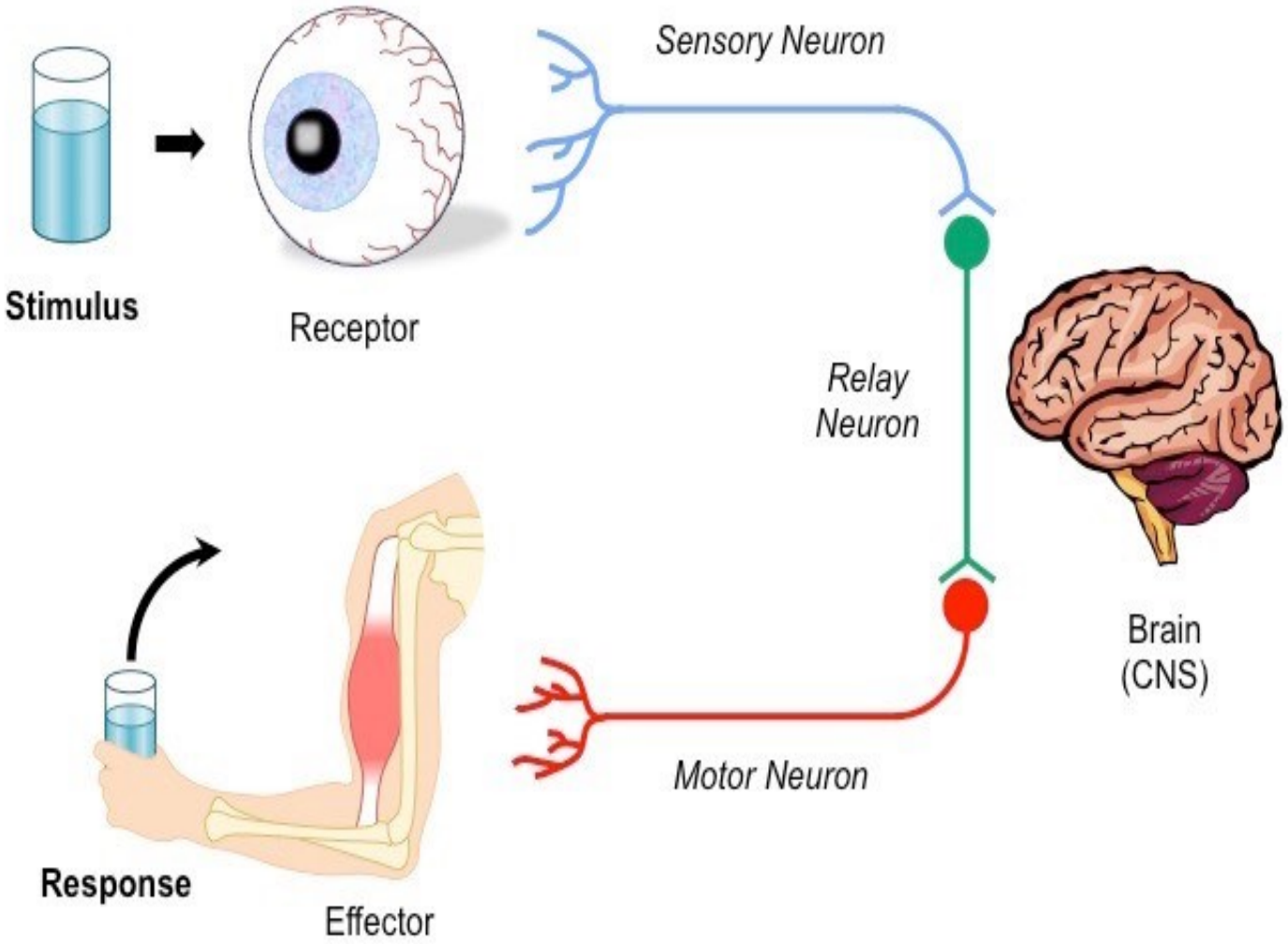
Neural network is inspired by biology,
In reality, it just a mathematical function

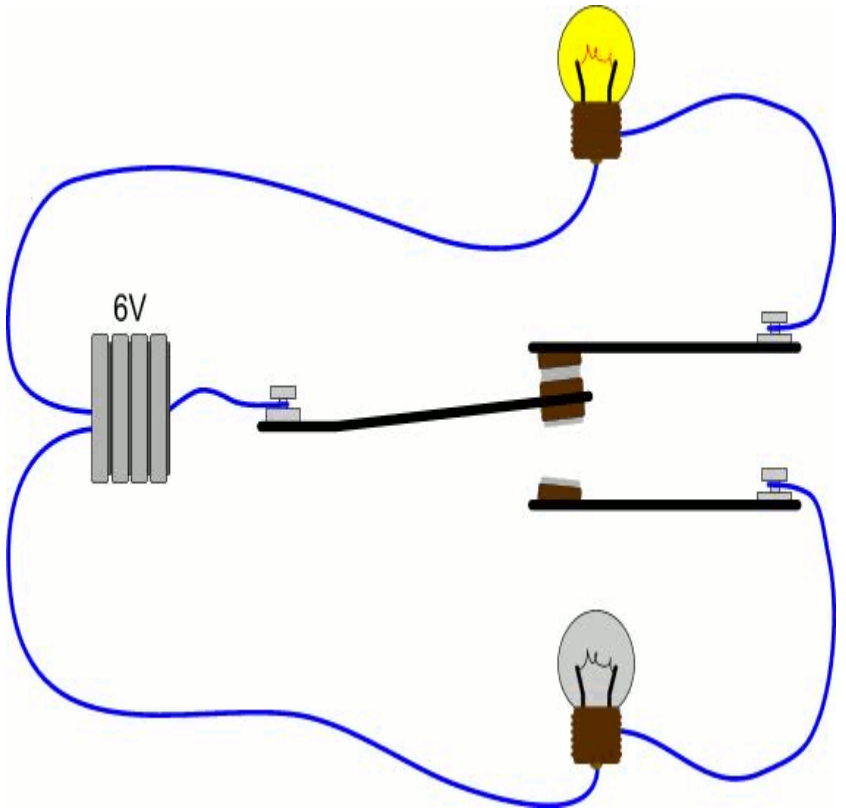


Neuron === Perceptron

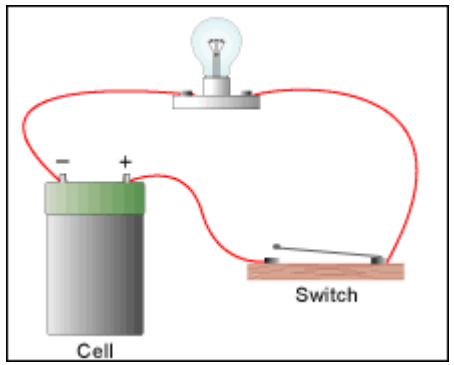


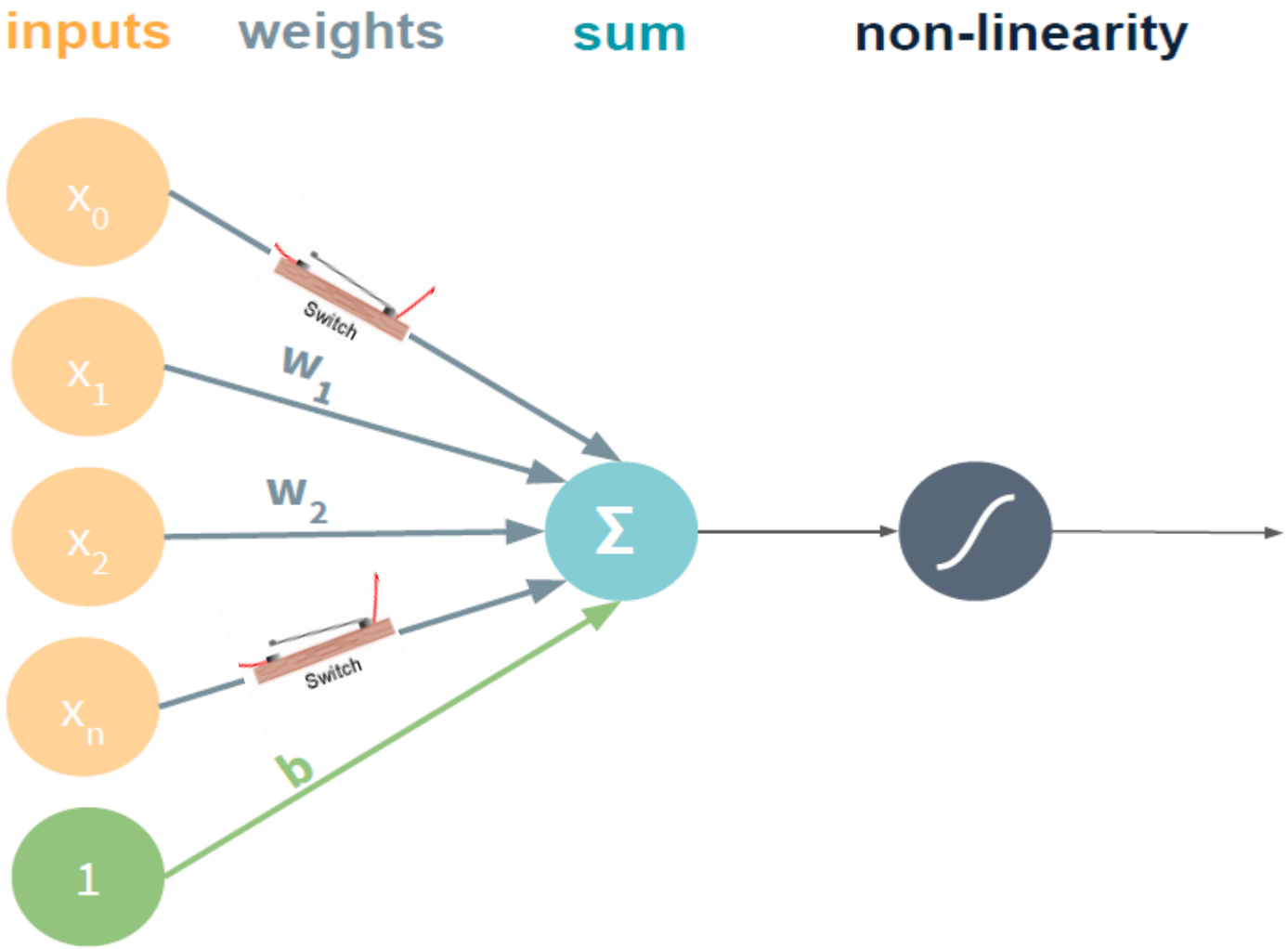
Switches On/Off a nerve connection





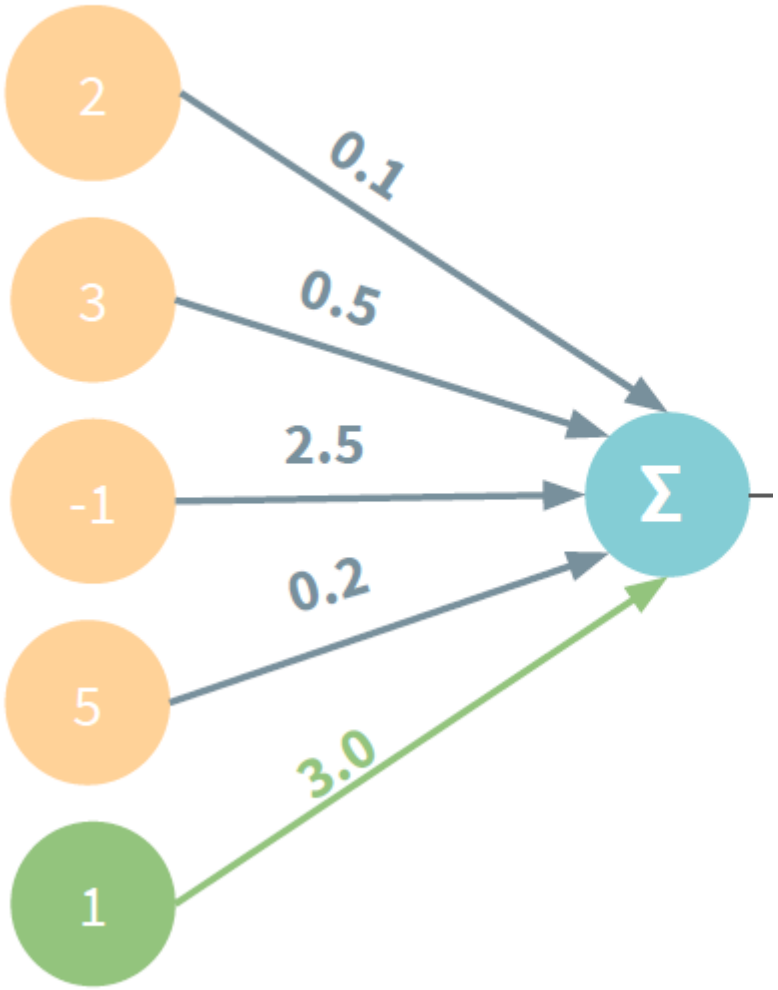
Switches On/Off a nerve connection





Simply an weighted average

- inputs weights sum



(2*0.1) +

(3*0.5) +

(-1*2.5) + output → Sum = 3.3

(5*0.2) +

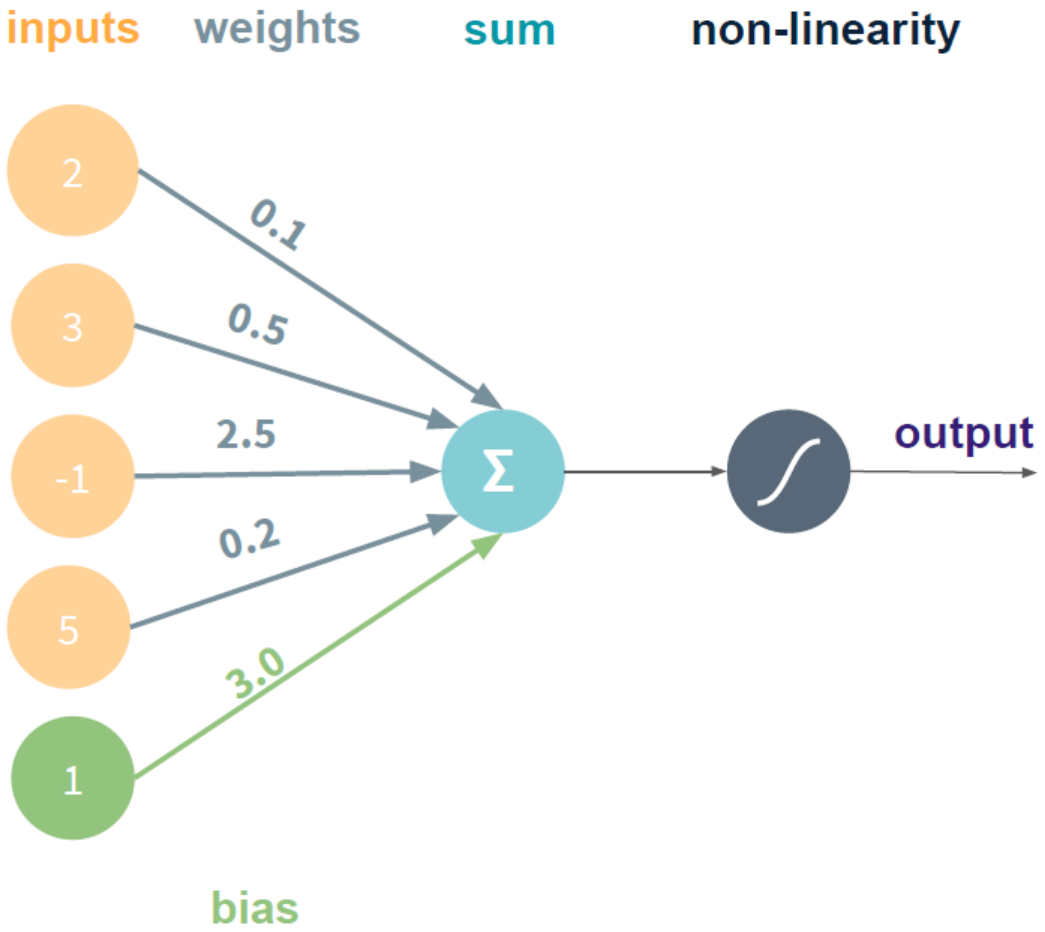
(1*3.0)

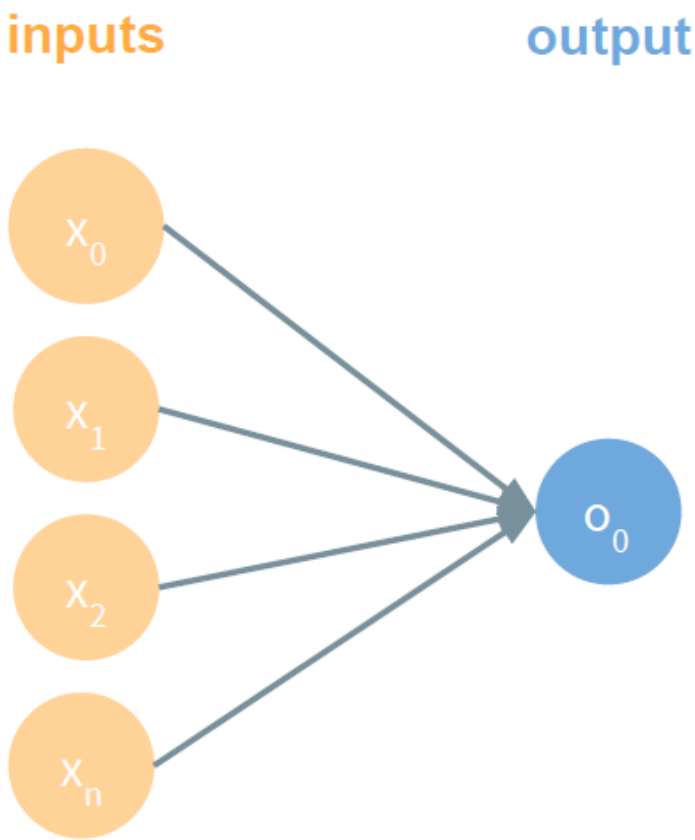
Then a "activation function"

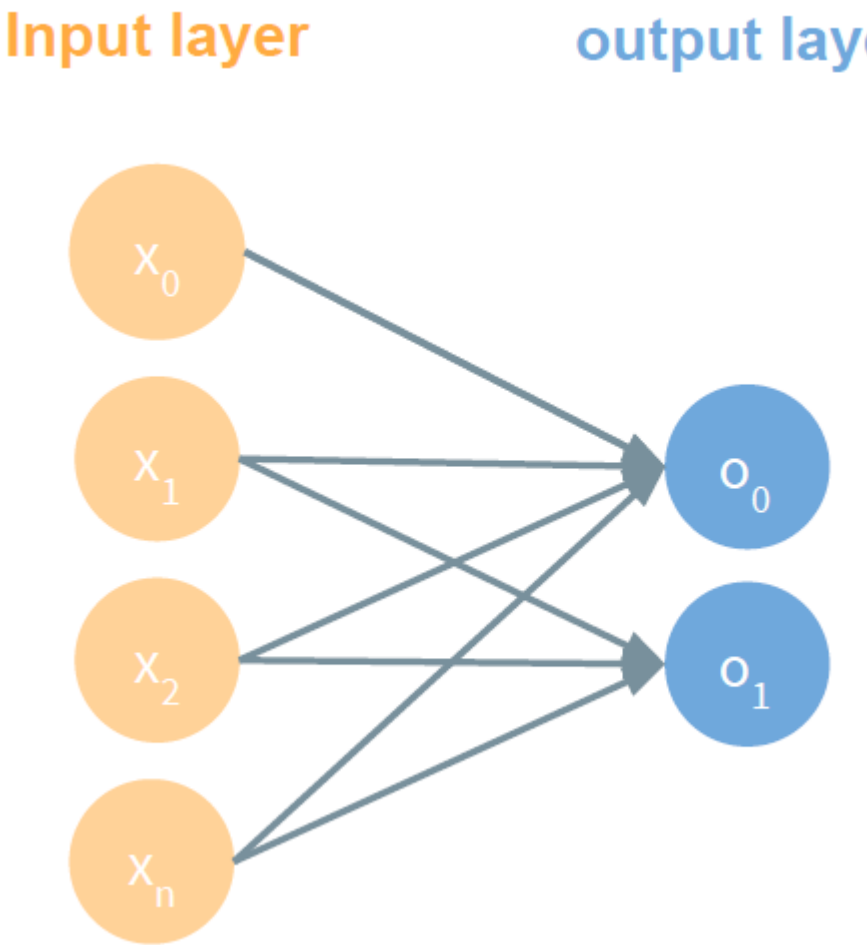
- Perceptron Forward Pass

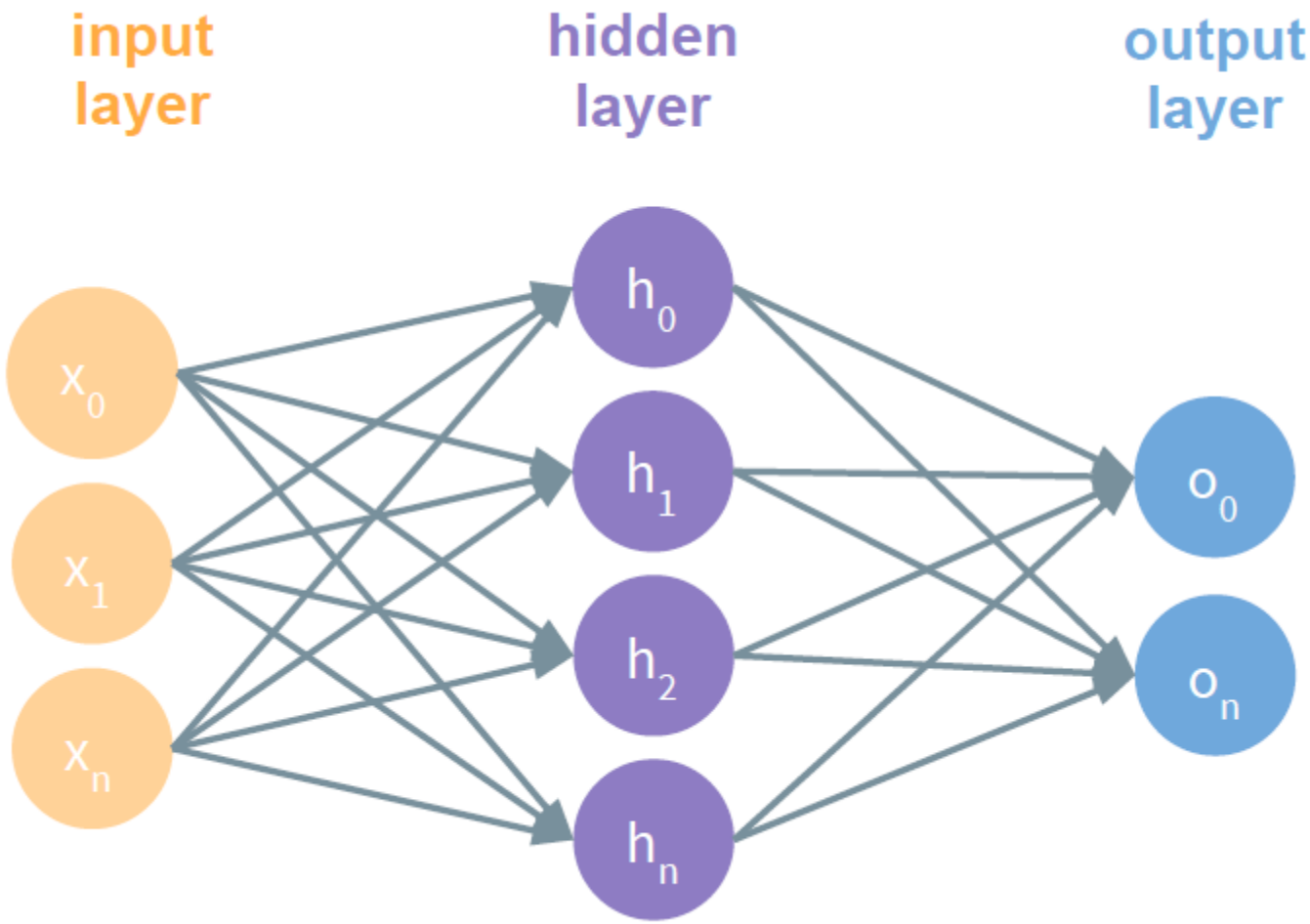
$$output = g(3.2) = \sigma(3.2)$$

$$= \frac{1}{(1 + e^{-3.2})} = 0.96$$

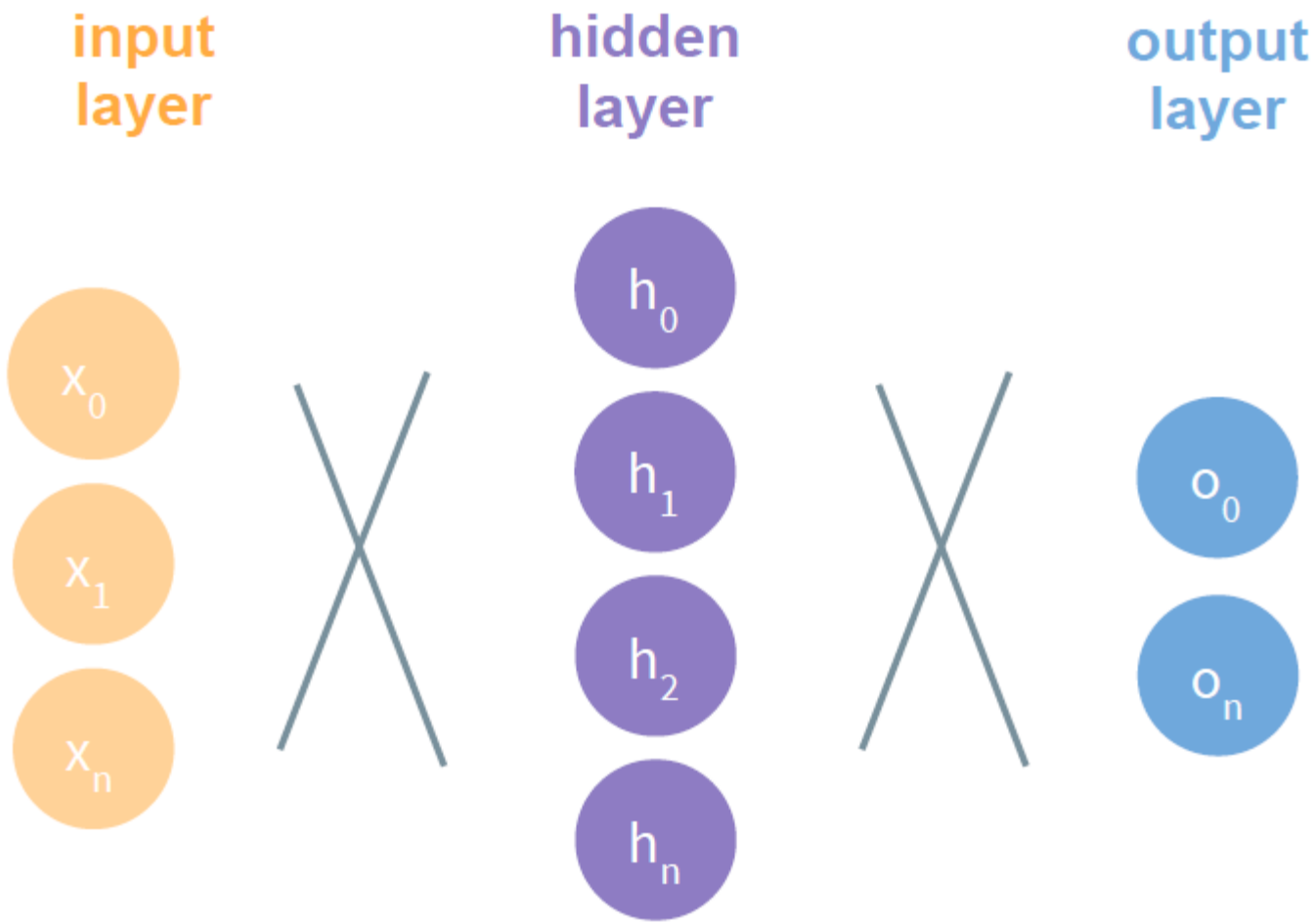



inputs
output
x_0
x_1
x_2
x_n
o_0

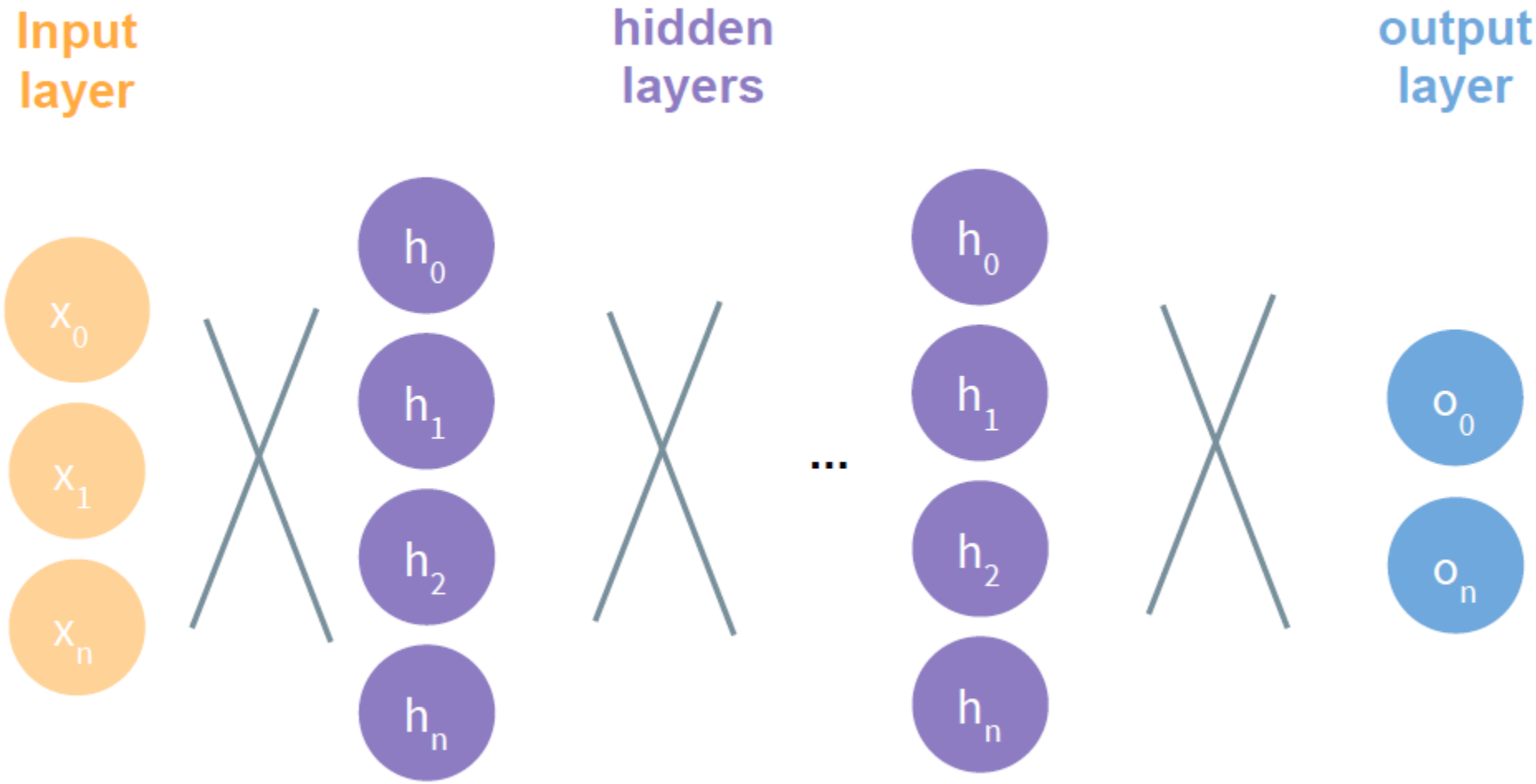

Input layer
output layer
x_0
x_1
x_2
x_n
o_0
o_1


input layer
hidden layer
output layer
x_0
x_1
x_n
h_0
h_1
h_2
h_n
o_0
o_n

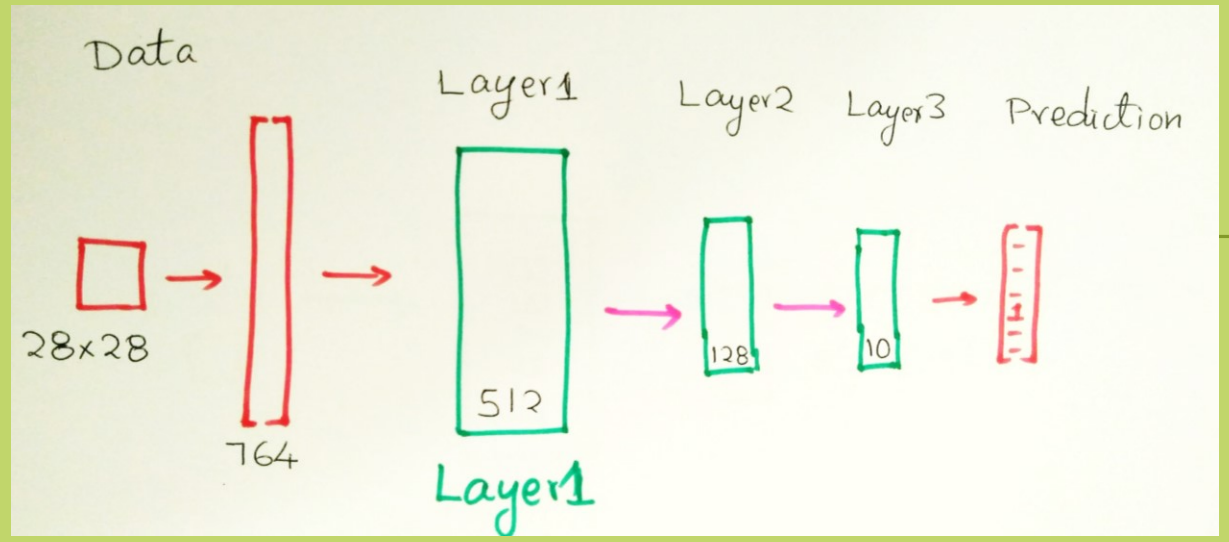
Multi-Layer Perceptron (MLP)

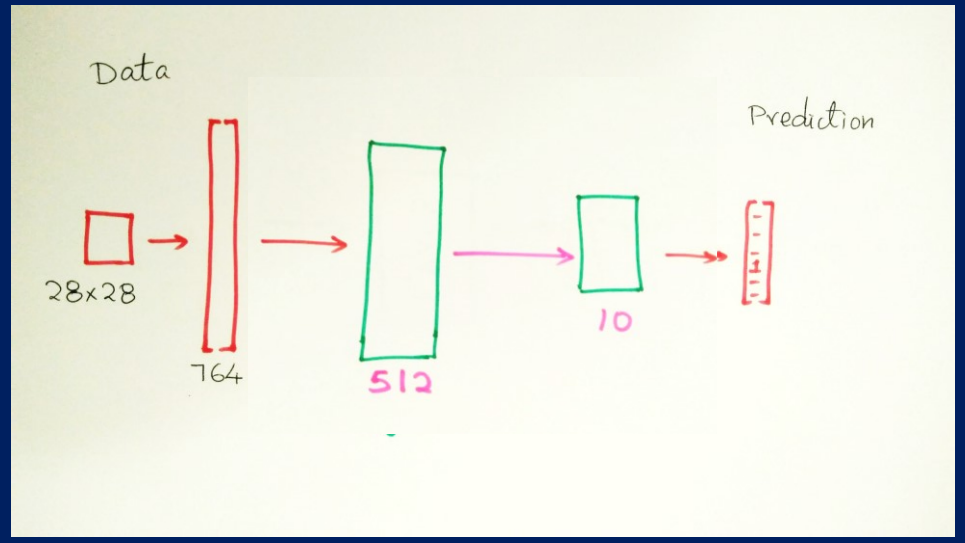


Deep Neural Network

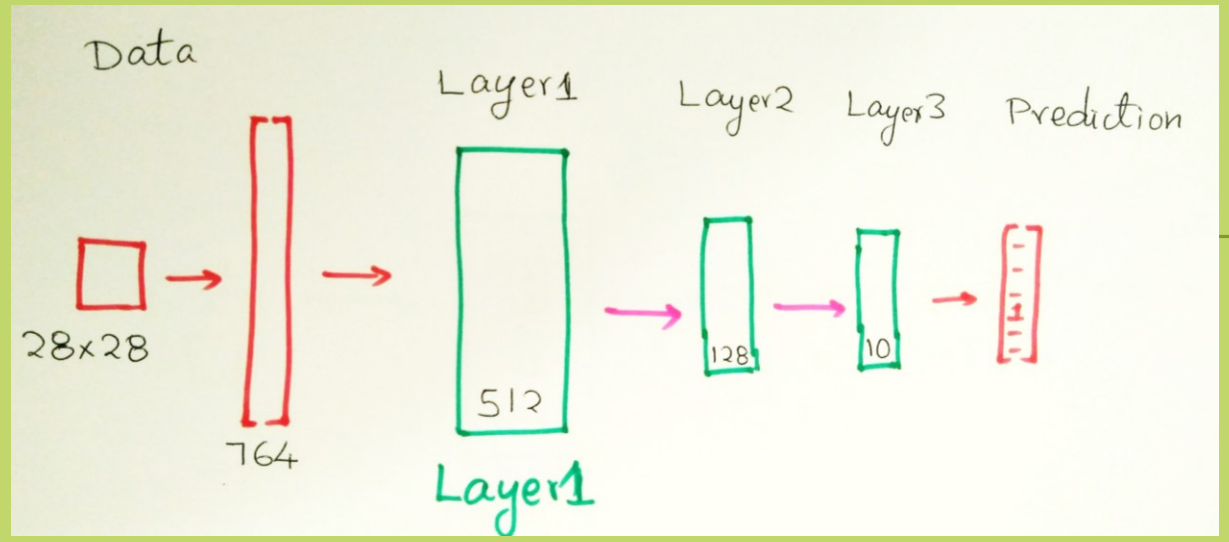


Design the network architecture

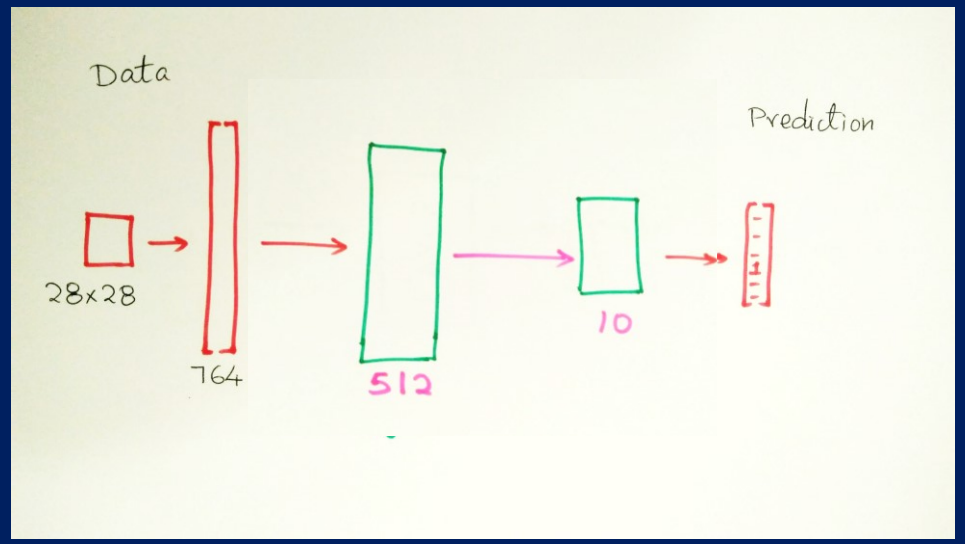




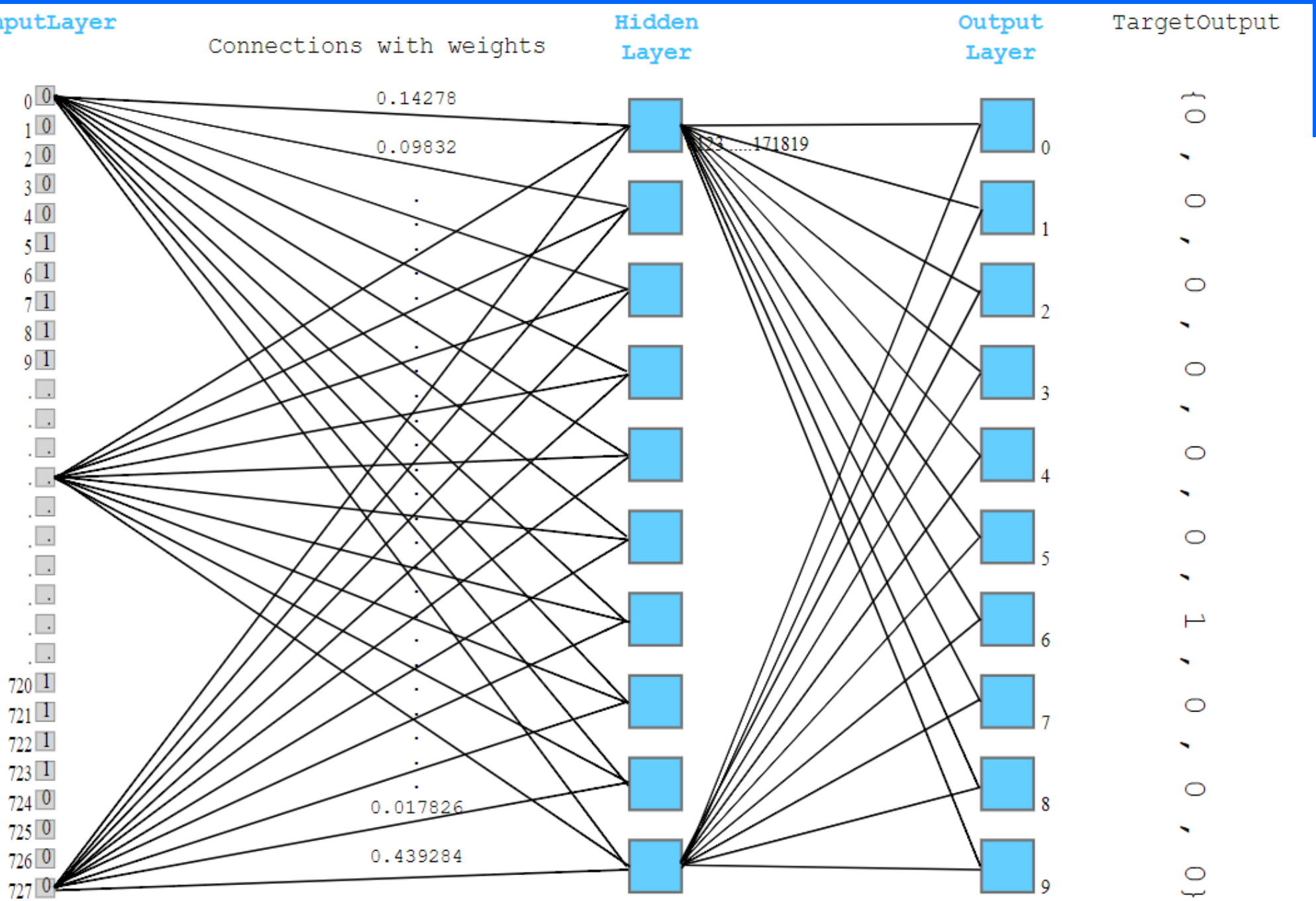
Design the network architecture




```
network.add( keras.layers.Dense(512))
network.add( keras.layers.Dense(128))
network.add( keras.layers.Dense(10))
```



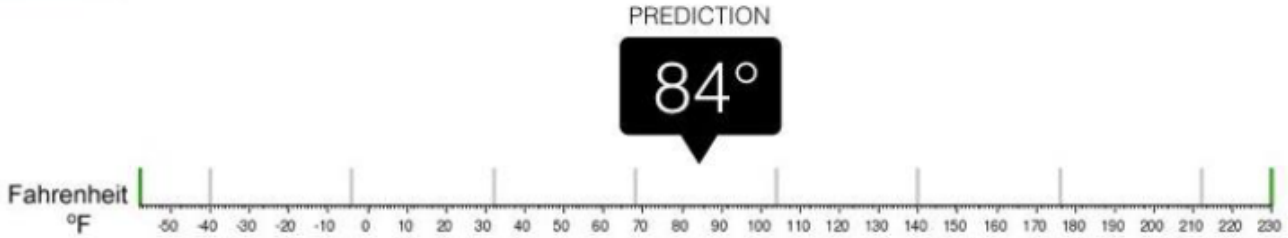

```
network.add( keras.layers.Dense(512))
network.add( keras.layers.Dense(10))
```



Regression vs Classification

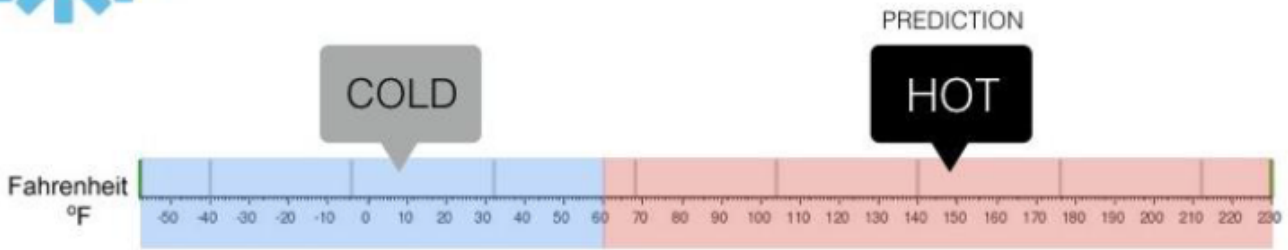
Regression

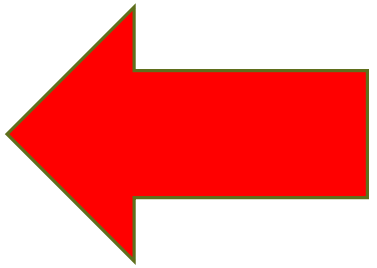
What is the temperature going to be tomorrow?



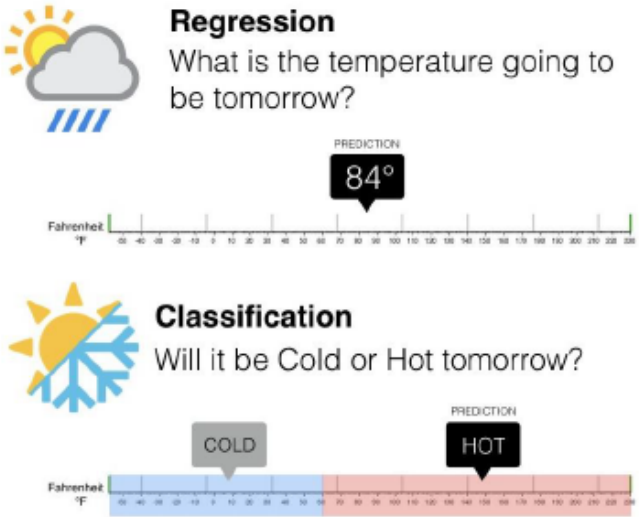
Classification

Will it be Cold or Hot tomorrow?





Loss Functions



- Loss function quantifies gap between prediction and ground truth
- For regression:
  - Mean Squared Error (MSE)
- For classification:
  - Cross Entropy Loss

Mean Squared Error

$$MSE = \frac{1}{N}\sum(t_i - s_i)^2$$

Prediction

Ground Truth

Cross Entropy Loss

$$CE = -\sum_{i}^{C} t_i log(s_i)$$

Classes

Prediction

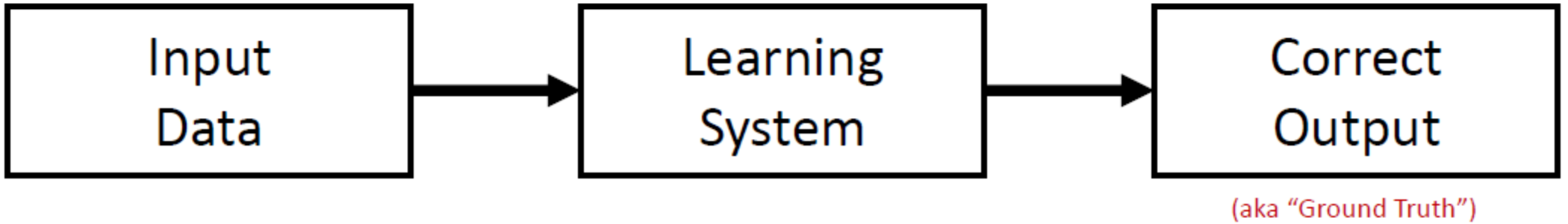
Ground Truth {0,1}

Keras: Loss Functions

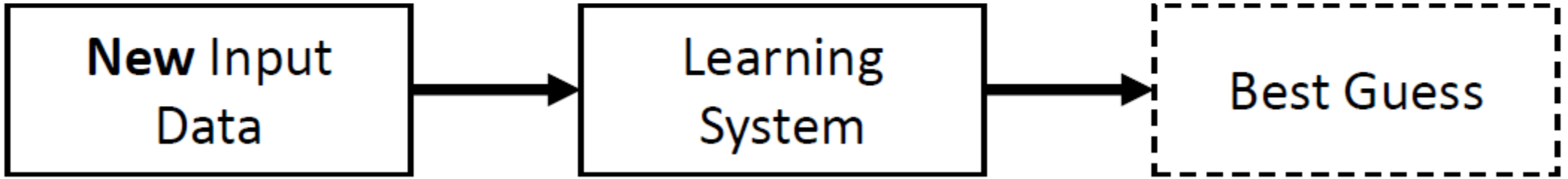
- Loss functions are used to compare the network's predicted output with the real output, in each pass of the backpropagations algorithm
  - Loss functions are used to tell the model how the weights should be updated
- Common loss functions are:
  - Mean squared error
  - Cross-entropy
  - etc.

Deep Learning: Training and Testing

Training Stage:

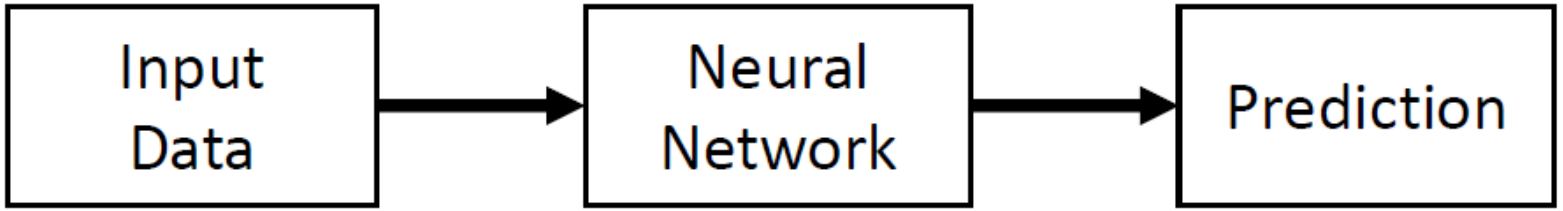


Testing Stage:

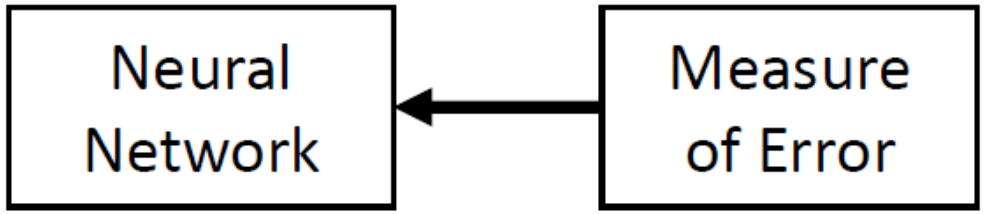


How Neural Networks Learn: Backpropagation

Forward Pass:

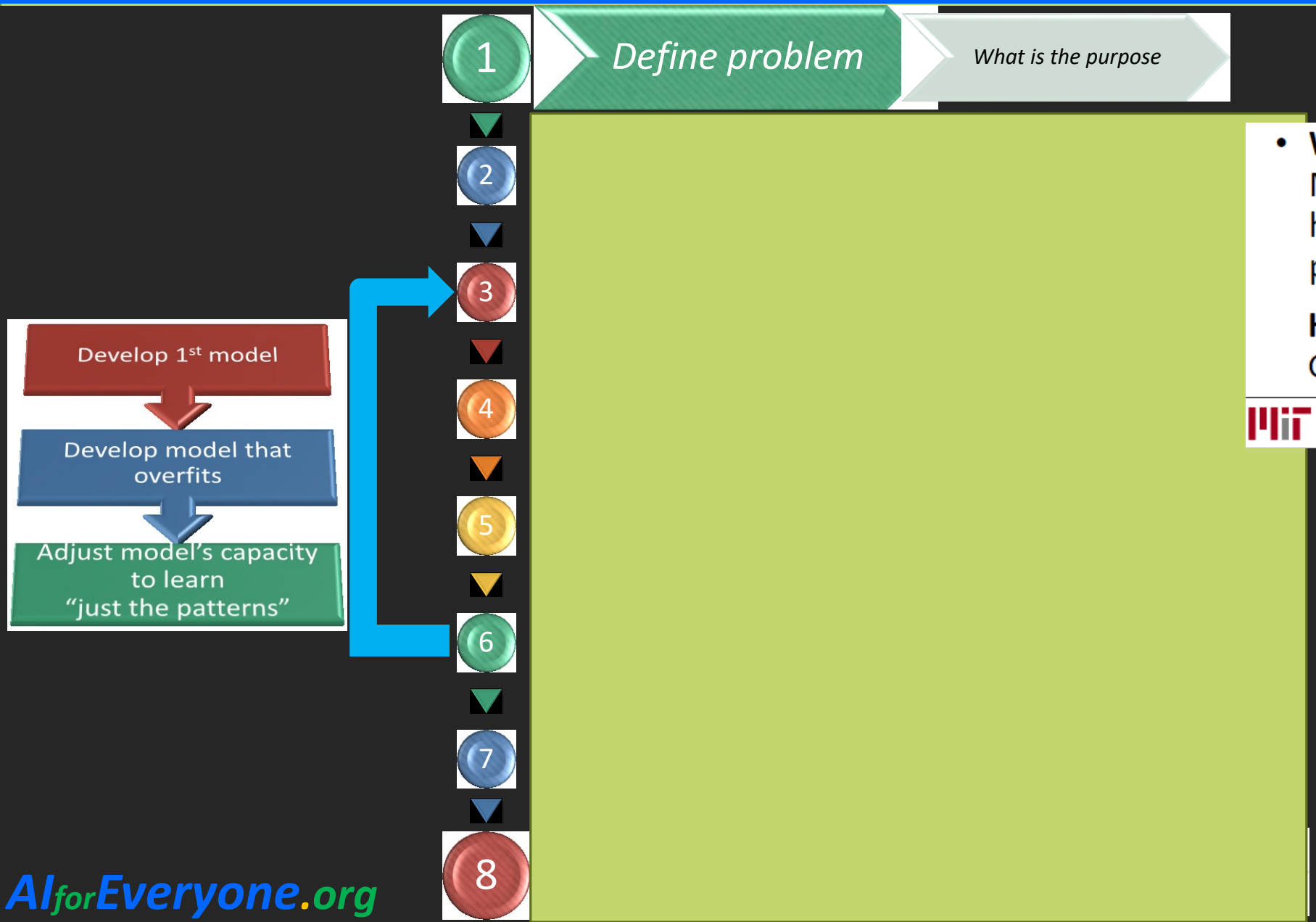


Backward Pass (aka Backpropagation):



Adjust to Reduce Error

Design of Deep Learning experiment



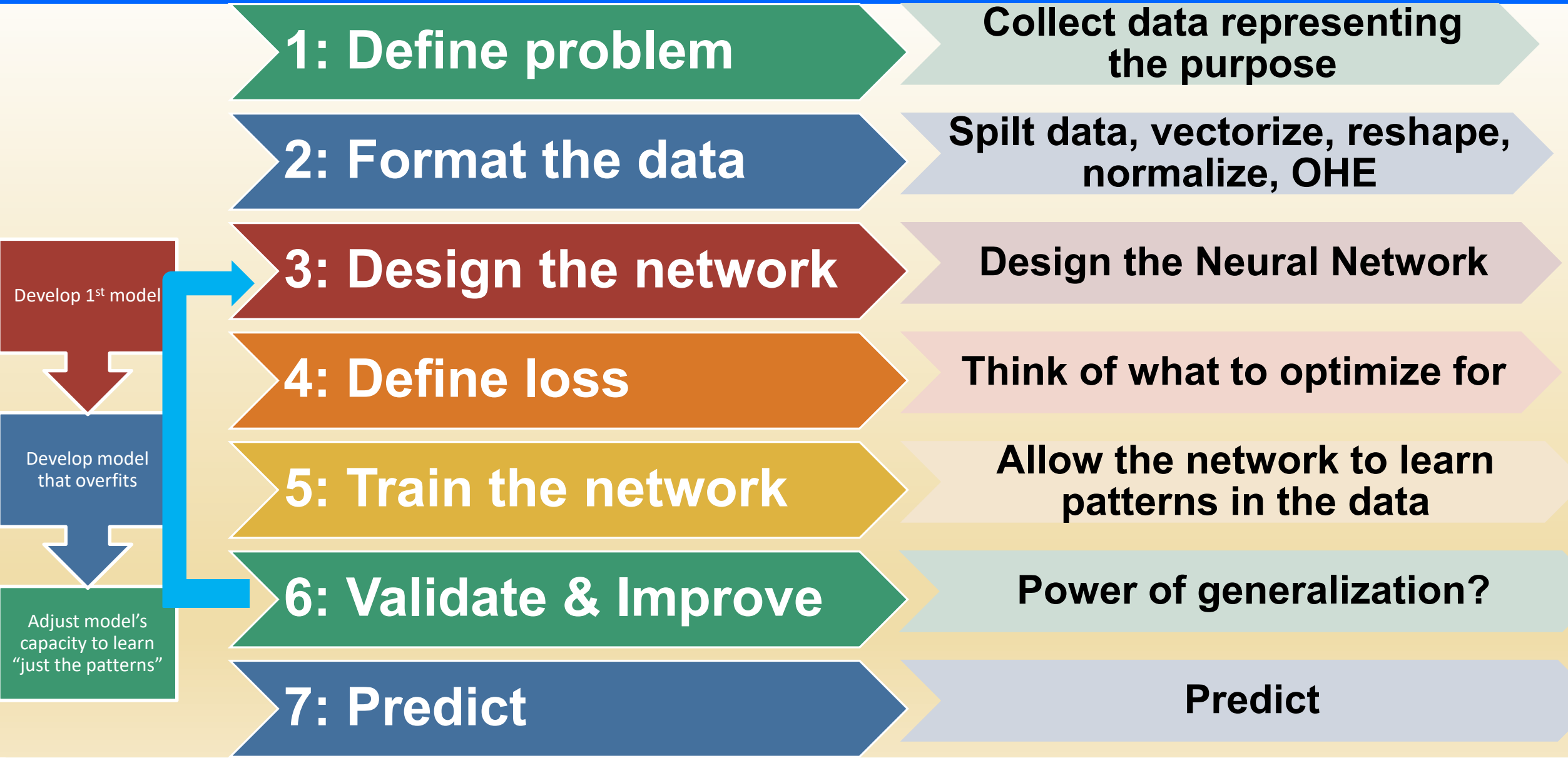
- **Where do we stand?**
  Most big questions of intelligence have not been answered nor properly formulated

Hard Part:
Good Questions + Good Data

Massachusetts Institute of Technology

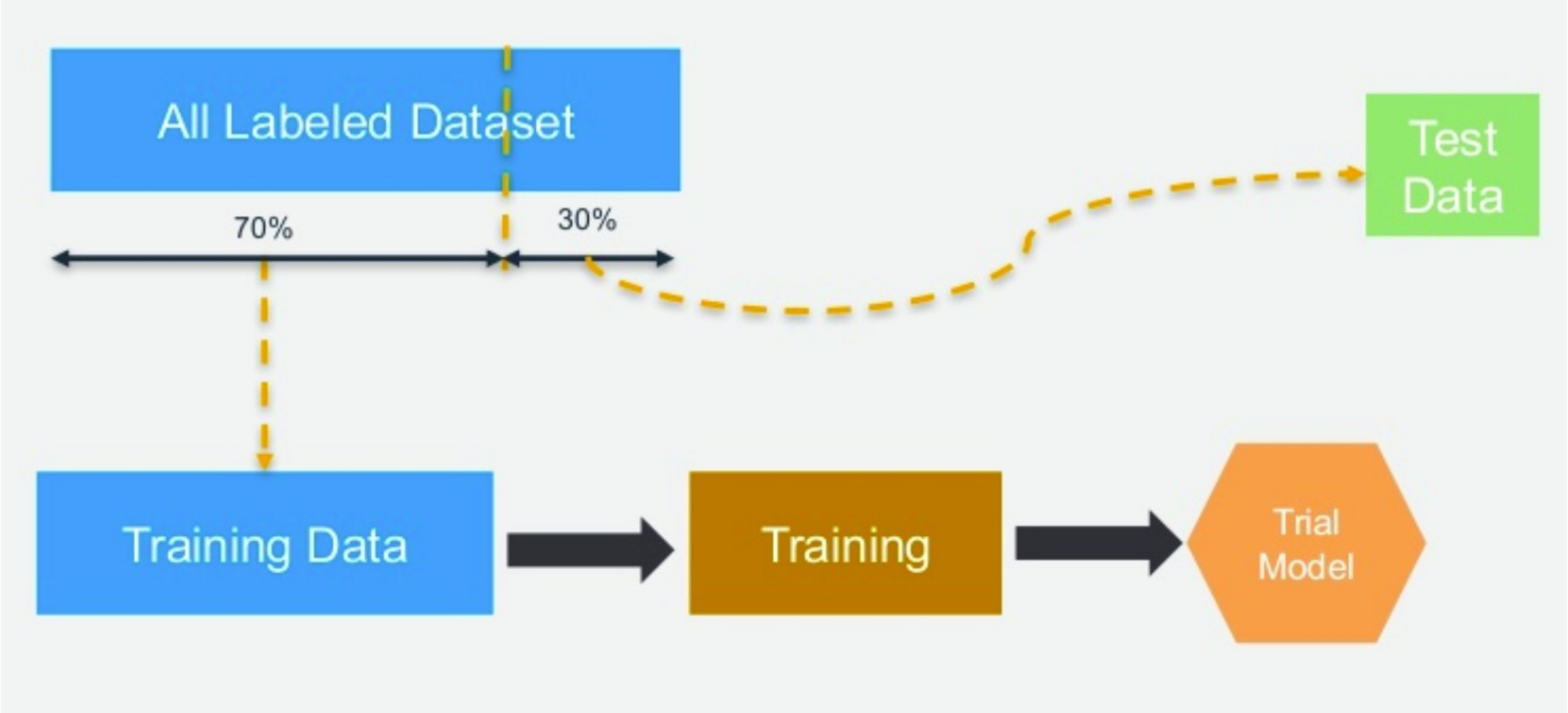
For the full list of references visit:
https://hcai.mit.edu/references

Method: 7 steps to develop your Deep Learning model



2. Prepare the data

Spilt data in training & testing | Vectortize as tensors | Reshape | Normalize | One Hot encode

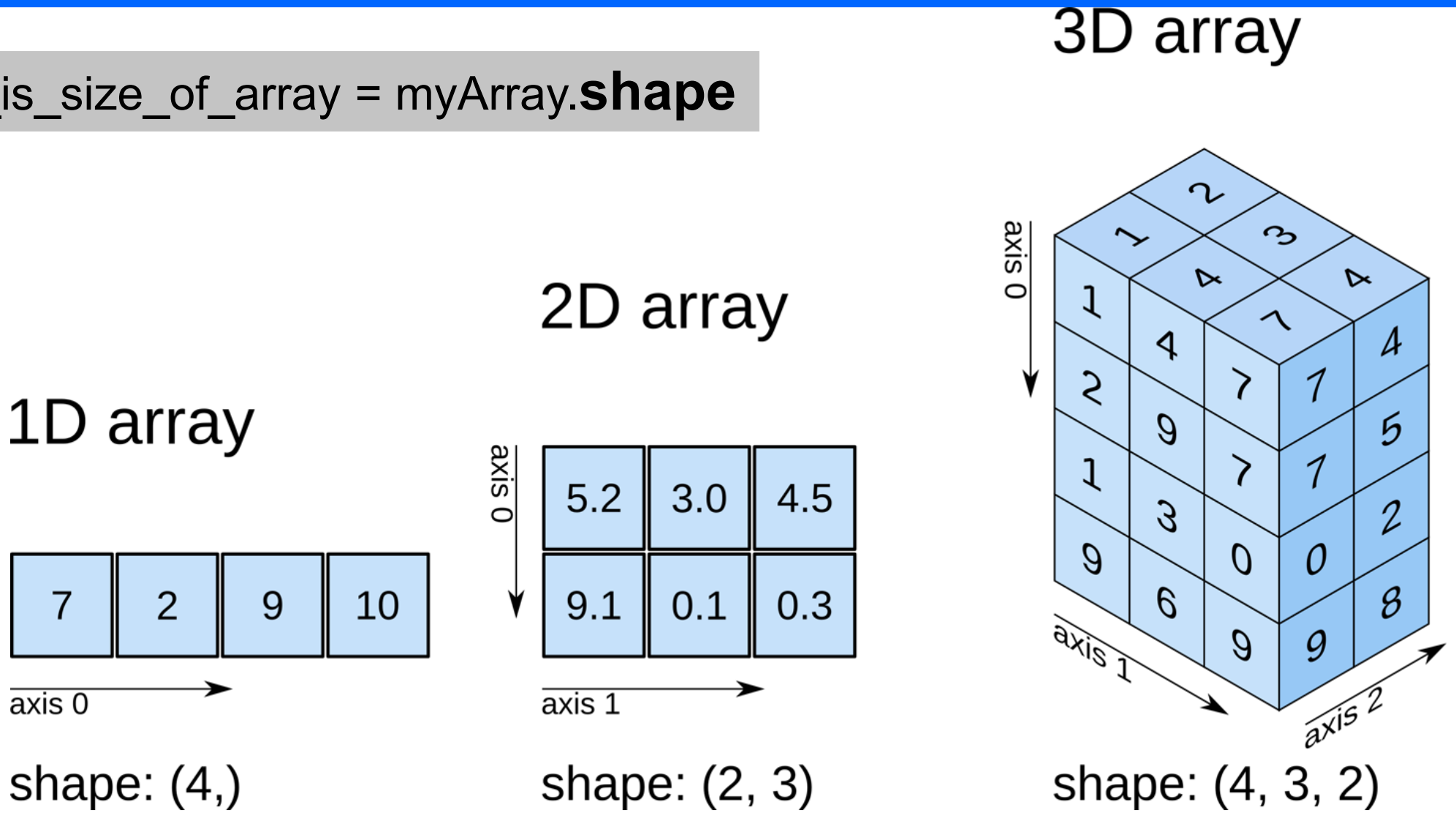


(trainX, trainY), (testX, testY) = mnist.load_data()

No of training records = 60000
No of test records = 10000

Multi dimensional Array == Tensor

What_is_size_of_array = myArray.**shape**



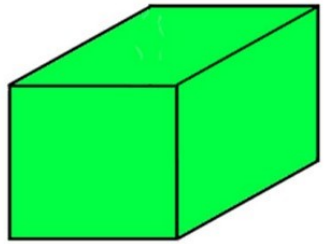
- 1D Tensor
  - Array
- 2D Tensor
  - 2D array
  - Matrix
- 3D Tensor
  - 3D array
- 4D Tensor
  -
- 5D Tensor

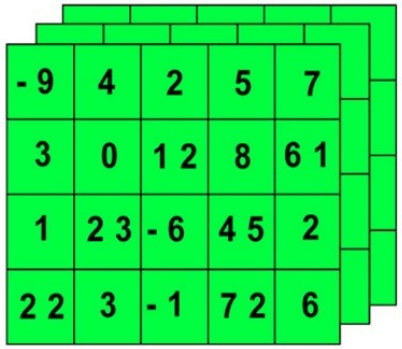
1D TENSOR/
VECTOR

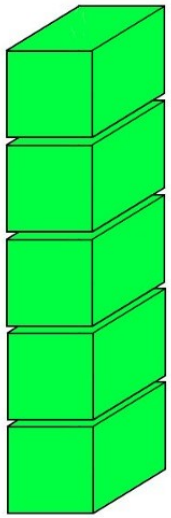
2D TENSOR /
MATRIX

- 9	4	2	5	7
3	0	1 2	8	6 1
1	2 3	- 6	4 5	2
2 2	3	- 1	7 2	6

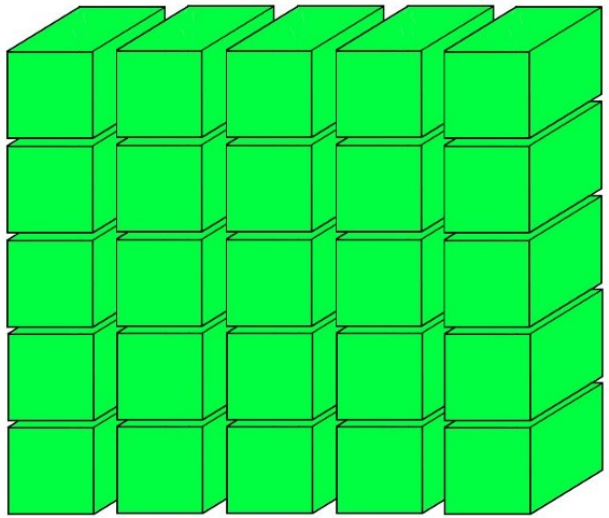
3D TENSOR/
CUBE





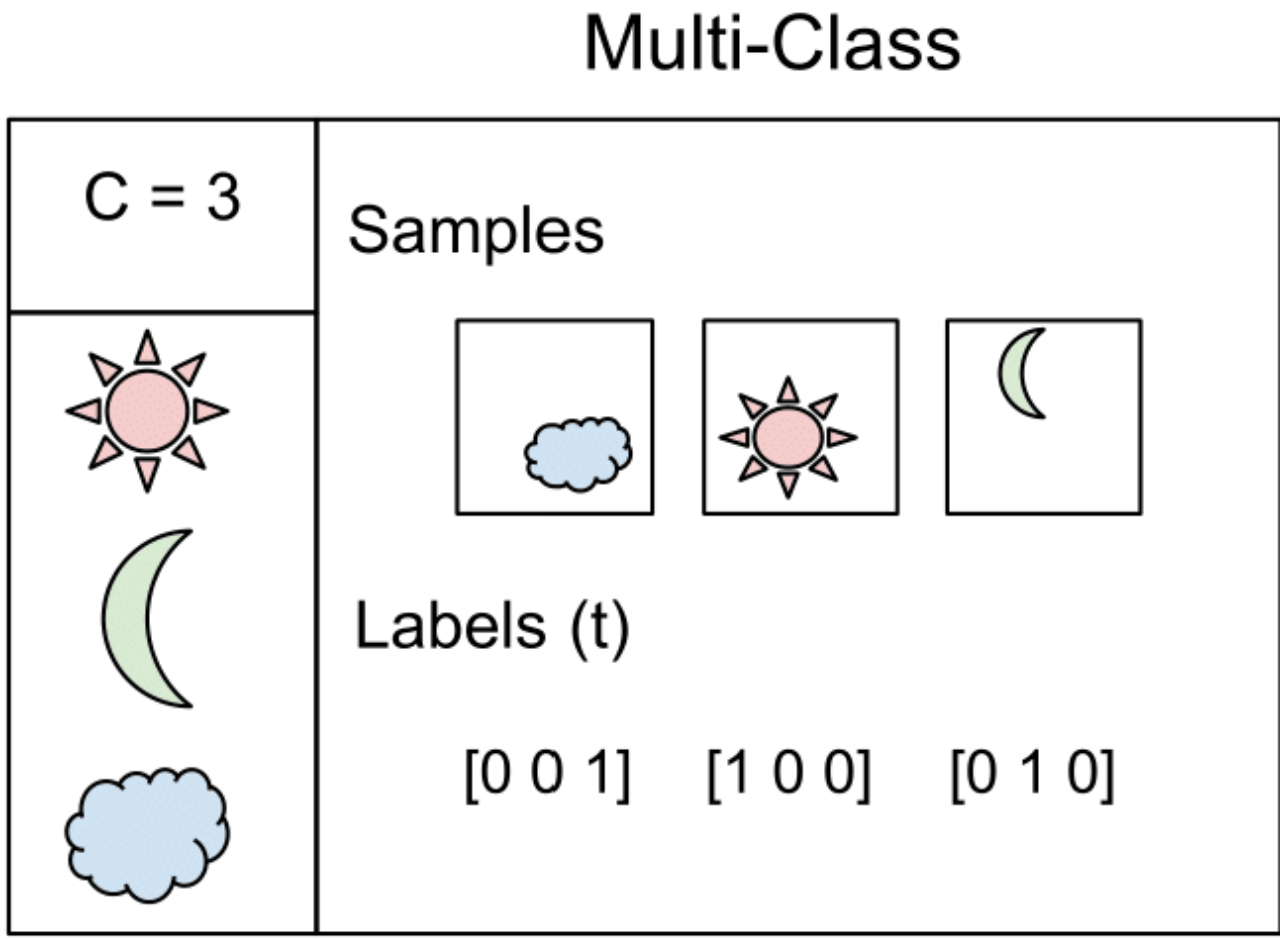


4D TENSOR
VECTOR OF CUBES



5D TENSOR
MATRIX OF CUBES

One Hot Encoding



yNew = to_categorical (y)

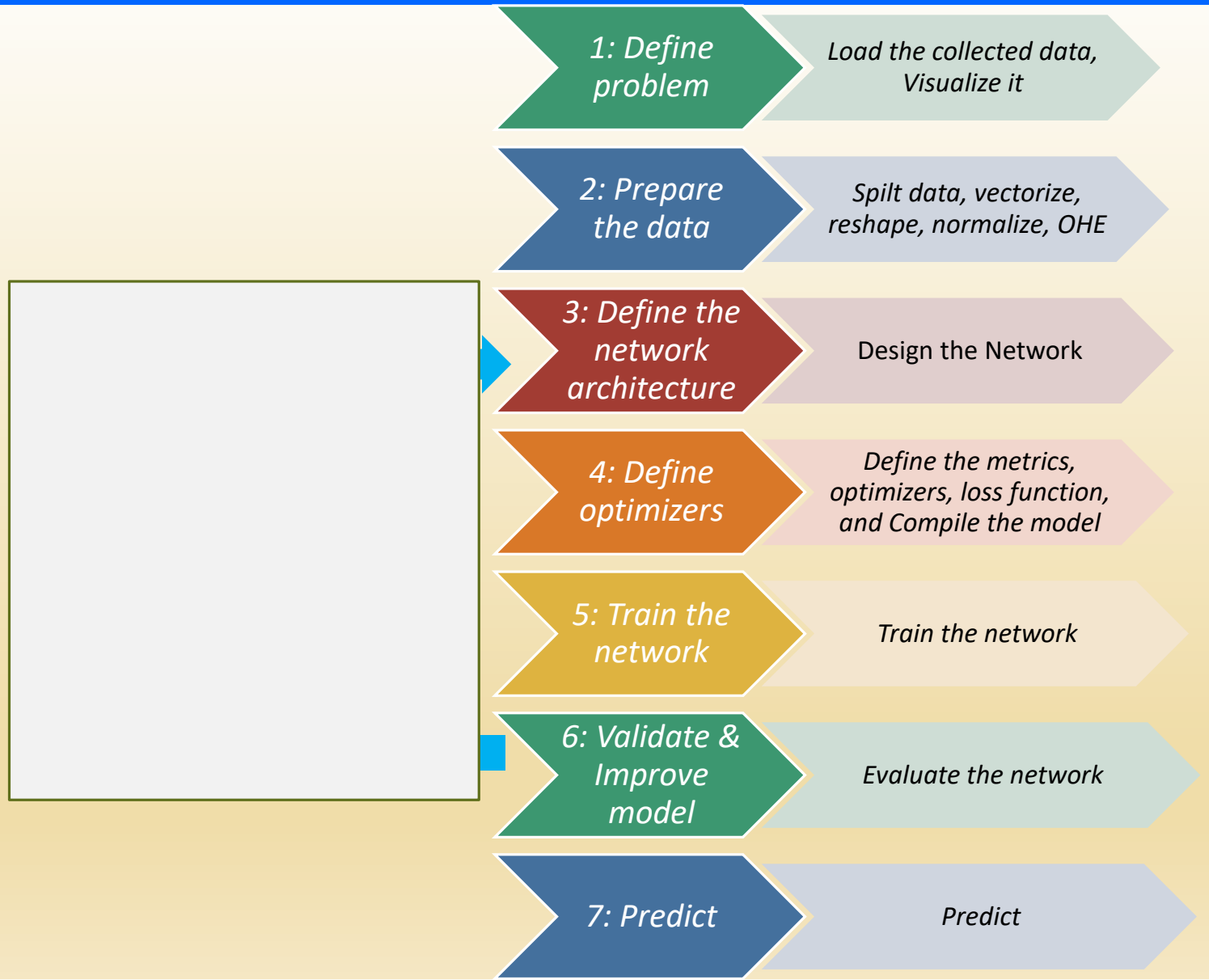
Data:
[2]

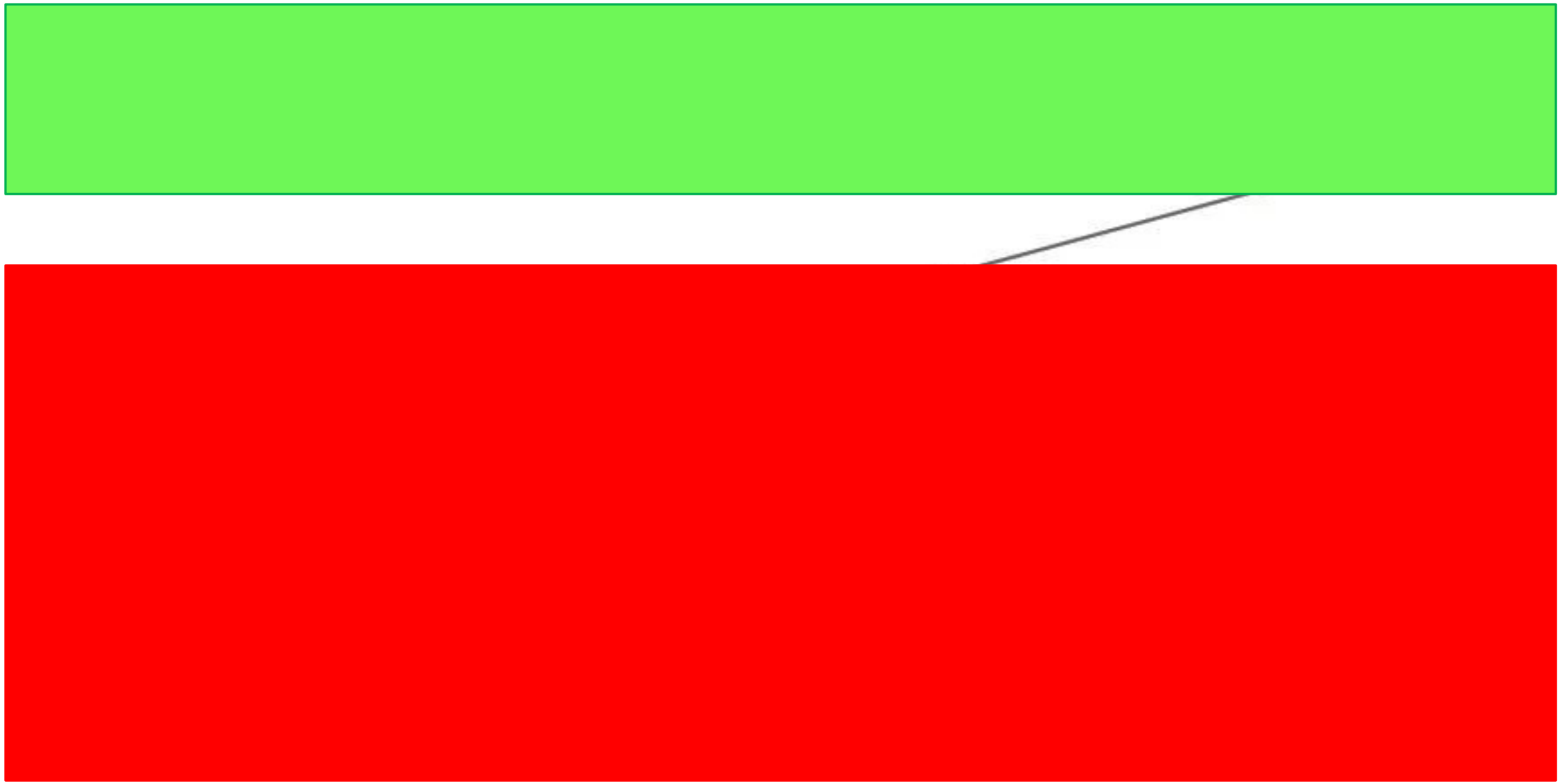
Data after **One Hot Encoding**:
[[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]]

myArray = keras.utils. to_categorical (aInteger)

Data: [2]
Data after OHE : [[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]]

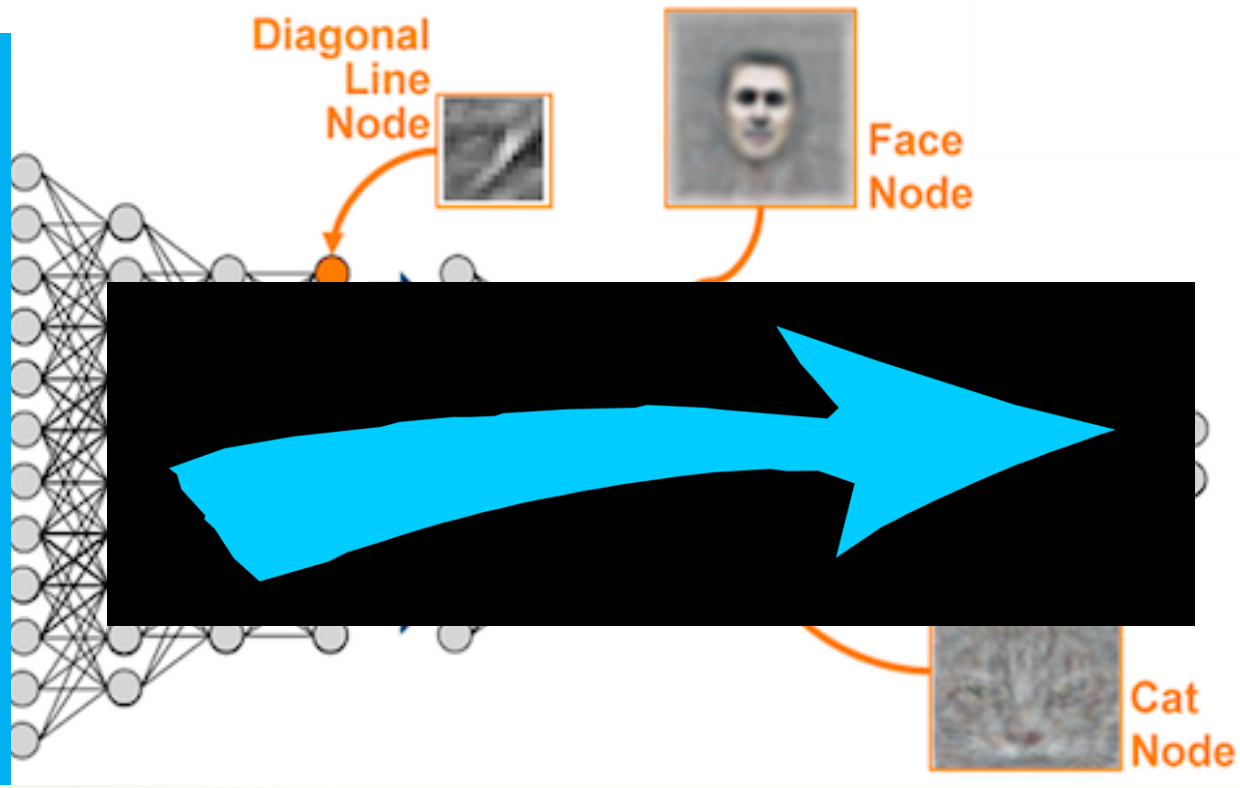
7 easy steps to develop your 1st Deep Learning model





3. Design the network

Data



Representation of Data

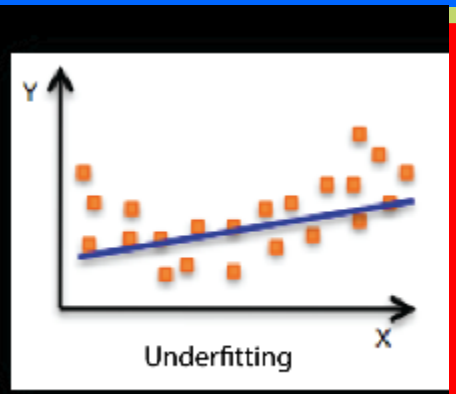
Develop 1st model

Develop model that overfits

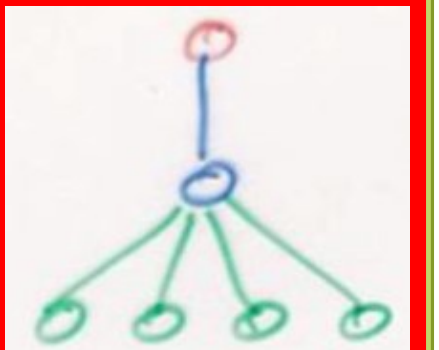
Adjust model's capacity to learn "just the patterns"

3. Design the network

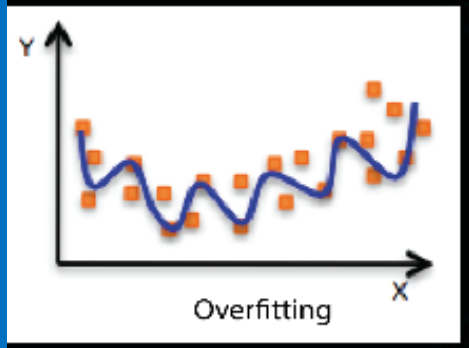
Develop 1st model



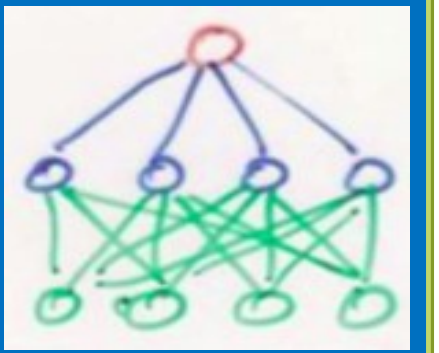
small capacity may prevent it from representing all structure in data



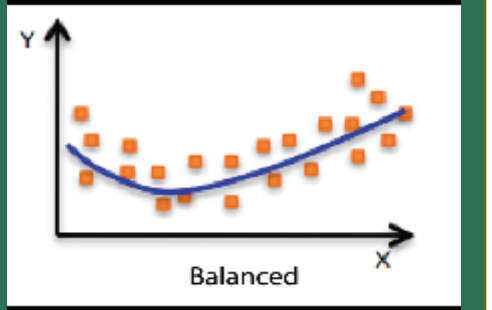
Develop model that overfits



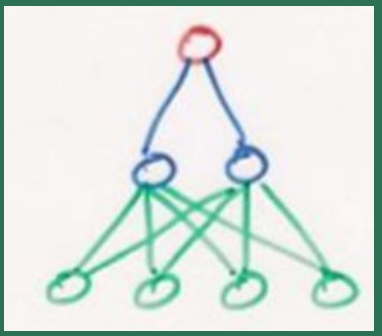
large capacity may allow it to memorize data and fail to capture regularities



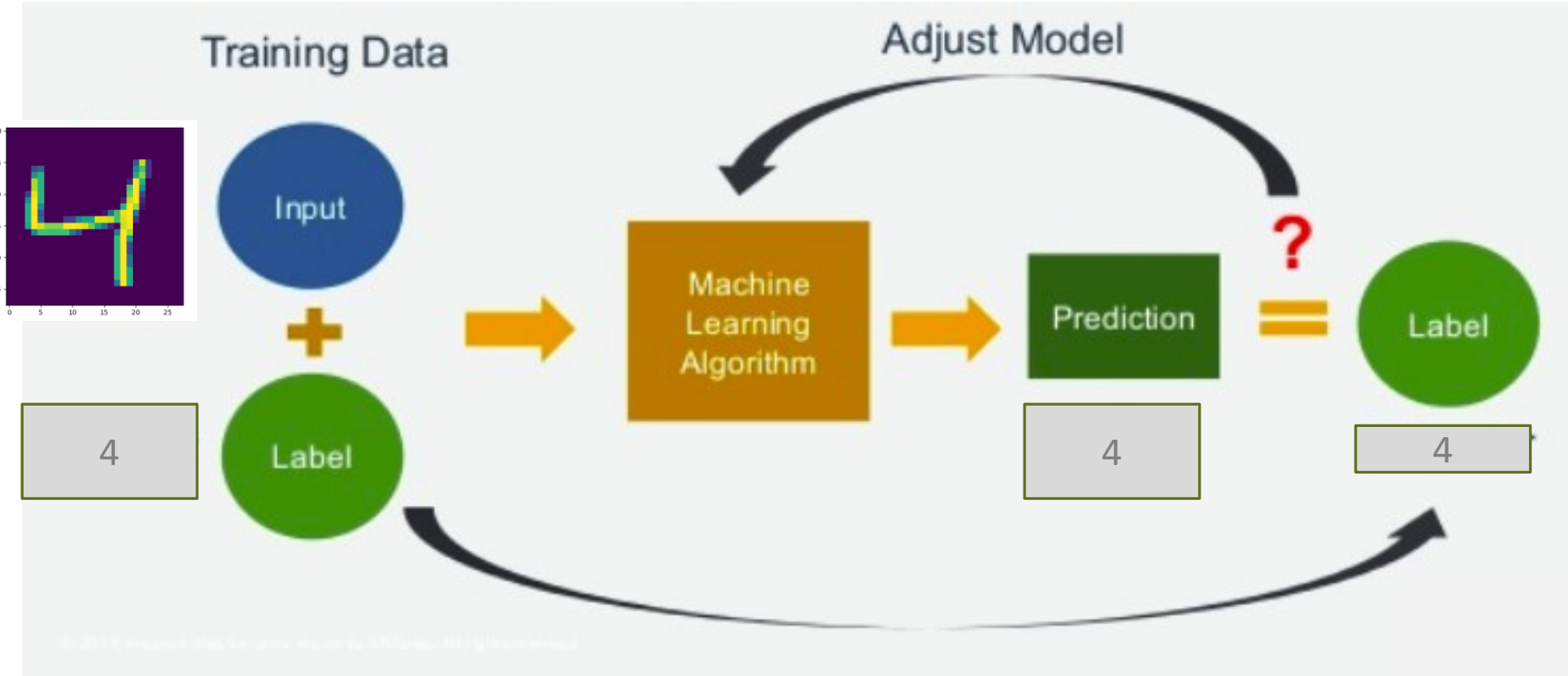
Adjust model's capacity to learn "just the patterns"



Limit the capacity so that only the representations can be learnt

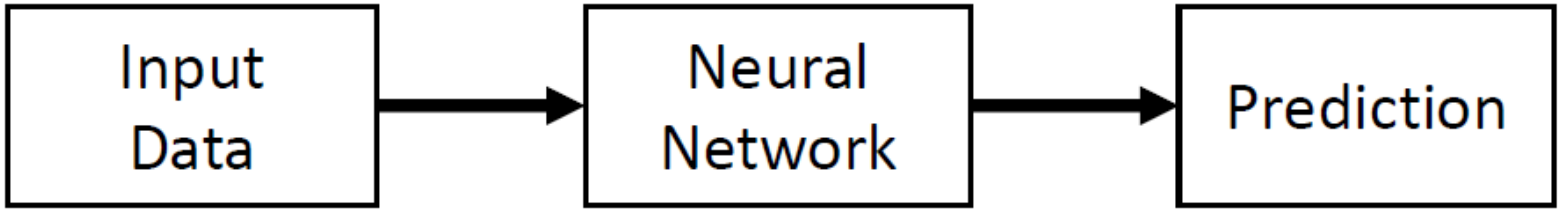


5. Train

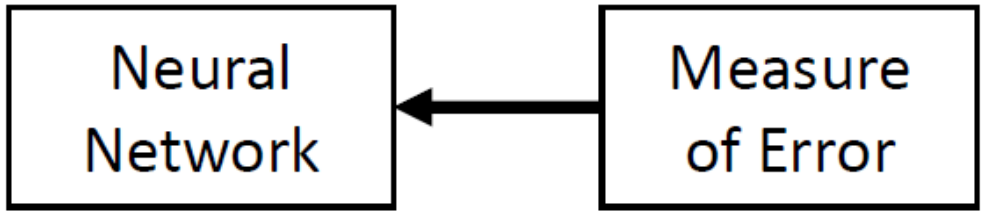

Training Data
Adjust Model
Input
Label
4
Machine Learning Algorithm
Prediction
4
?
Label
4

How Neural Networks Learn: Backpropagation

Forward Pass:

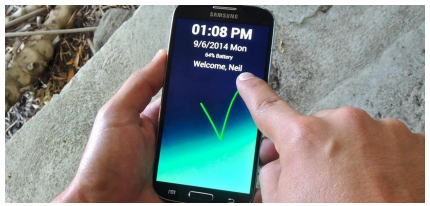


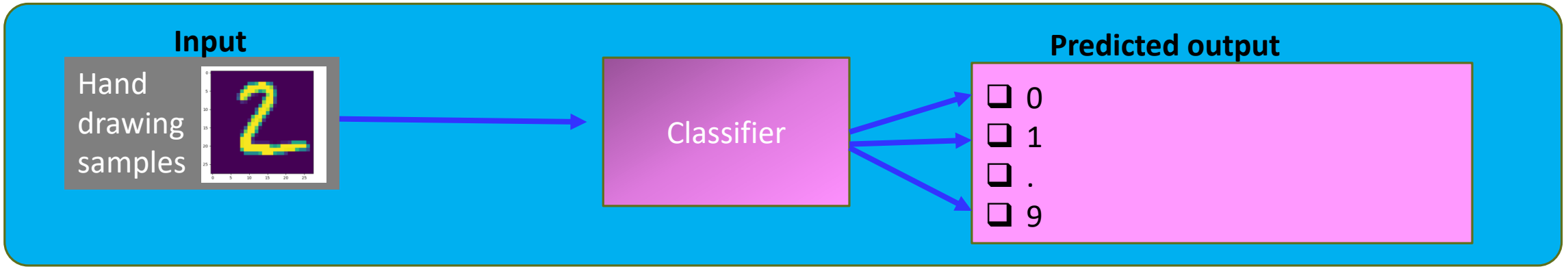
Backward Pass (aka Backpropagation):



Adjust to Reduce Error

7. Predict

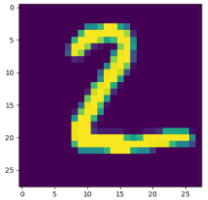

01:08 PM
9/6/2014 Mon
Welcome, Neil


Input
Hand drawing samples
Classifier
Predicted output
0
1
.
9

7. Predict

Problem: Users draws a hand drawn gestures , your app predicts it.

predictedResults = network.predict(testXready[1:3])



Predicted Results:

[1.35398892e-08 3.91871922e-08 9.99876499e-01 1.22501457e-04
1.77109036e-10 1.30937892e-07 2.99046292e-07 2.58880729e-11
4.58700157e-07 2.09212196e-11]

Ground Truth : [2]

Ground Truth OHE : [[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]]

Let’s it together

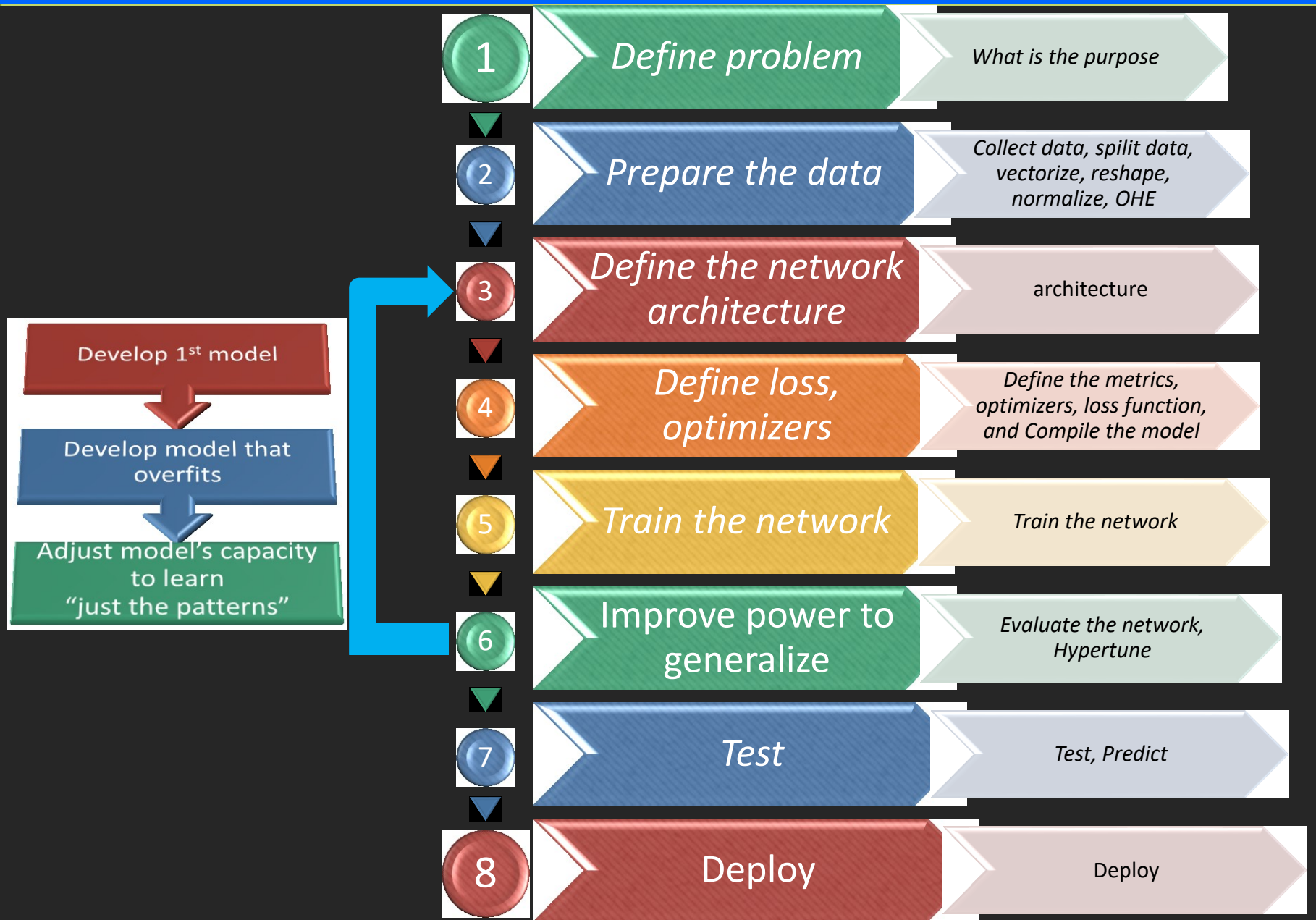
- App idea: Recognize your hand gesture to unlock your smartphone (Secure unlock)
- Biz outcome: Recognize hand draw gestures to unlock your phone

#Step 6: Evaluate the network

metrics_test_loss, metrics_test_accuracy = network.evaluate(testXready, testYready)

Test accuracy is: 0.9157

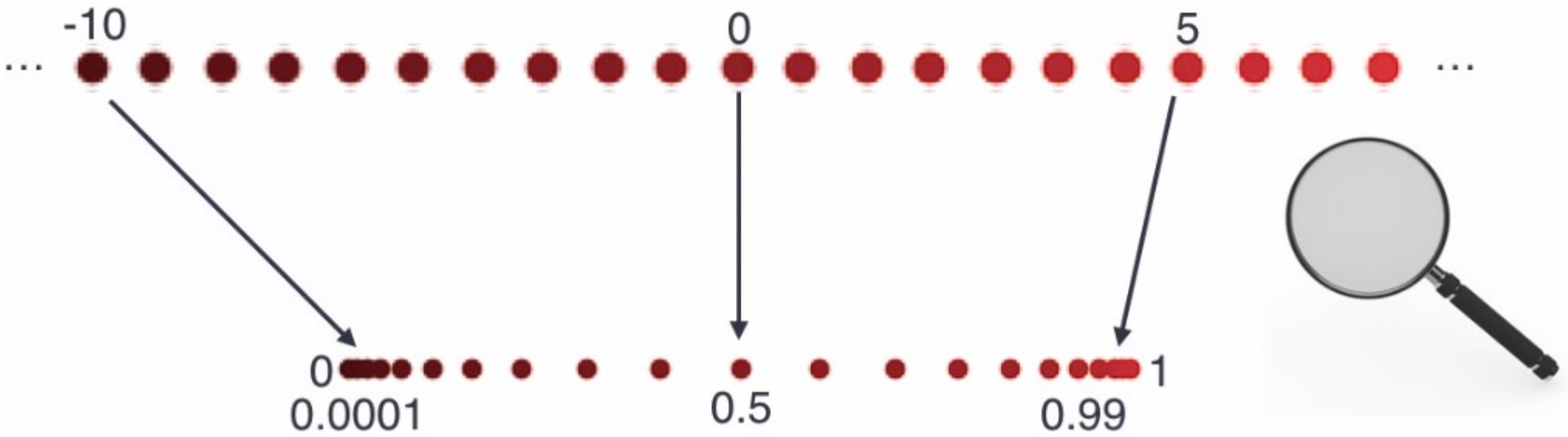
Design of Deep Learning experiment



Experiment 2

- MNIST_Predict_with_Dense_and_Tune_Capacity

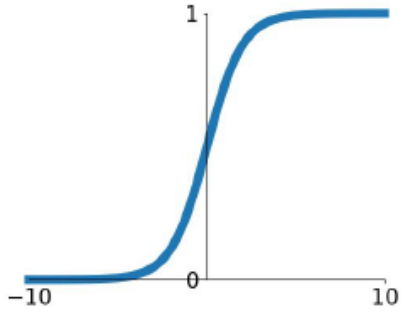
Activation function



Activation functions

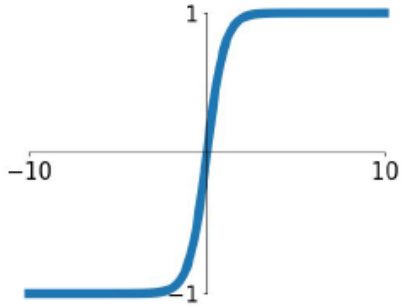
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



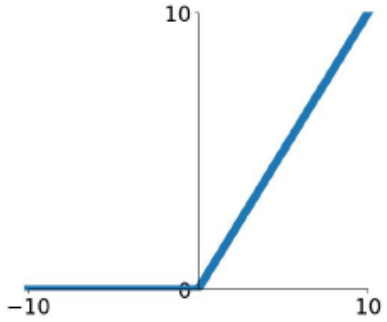
tanh

$$\tanh(x)$$



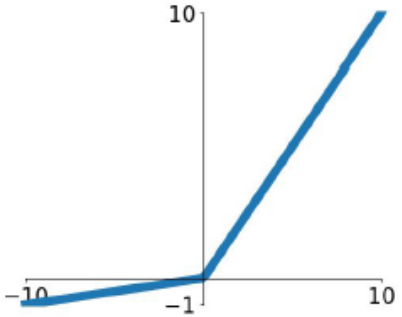
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

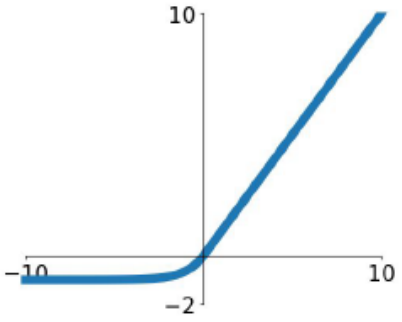


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

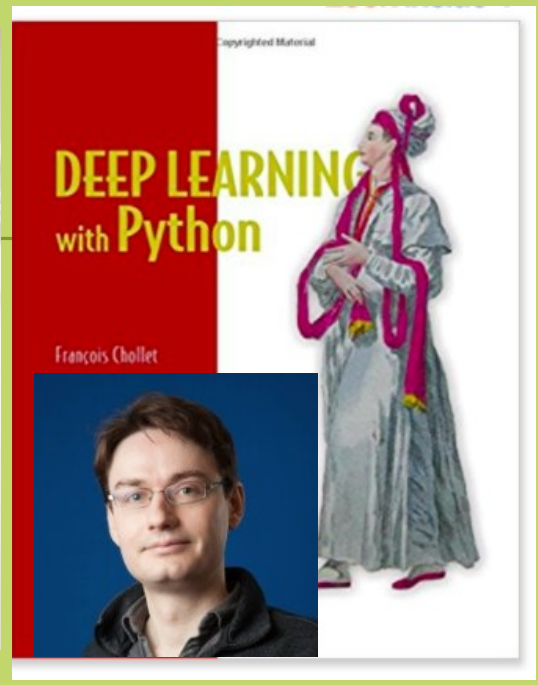
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Our inspiration

Friendly approaches :



1) KERAS.io

François Chollet's
Book on "Deep Learning with Python"

2) Deeplearning.ai (Coursera.org)

Andrew Ng

3) Google Machine Learning Course

https://developers.google.com/machine-learning/crash-course/

Acknowledgments & Credits are mentioned to inspirational resources presented in the end of presentation. For more like this, https://sites.google.com/view/AIforEveryone

More inspiration

Excellent Resources

- **Stanford cs231 n**

http://cs231n.stanford.edu

- **MIT Deep Learning**

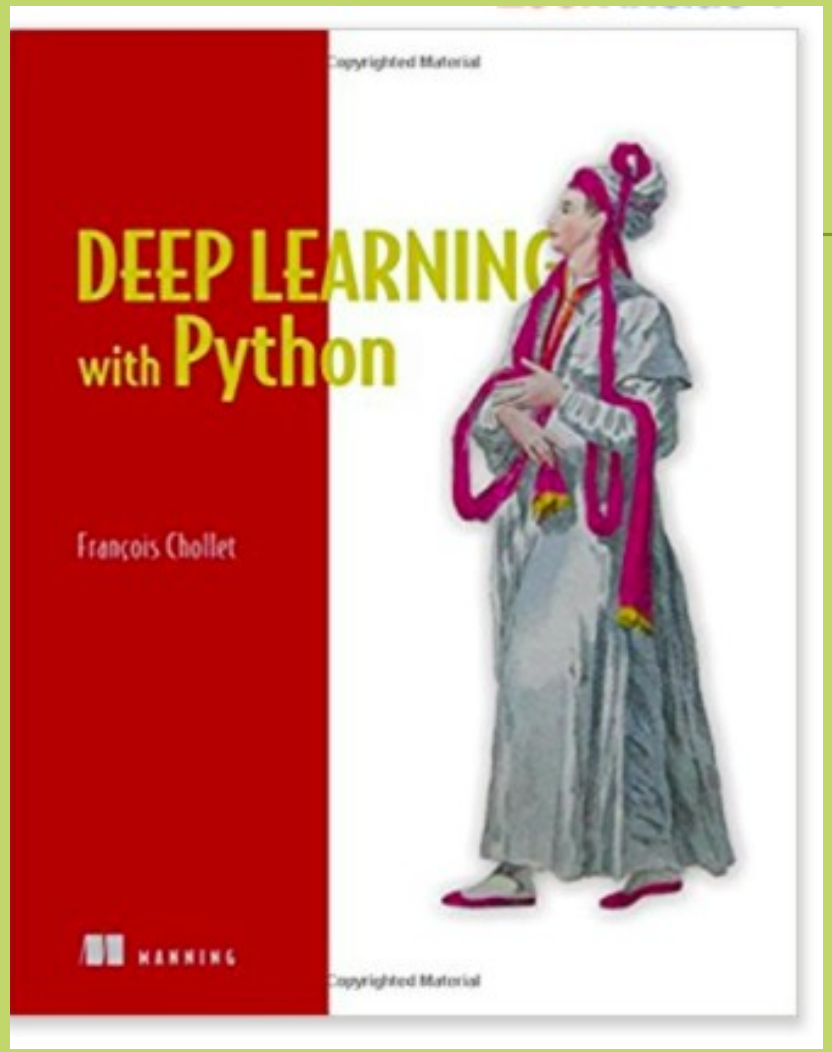
http://introtodeeplearning.com/
https://deeplearning.mit.edu

- **IIT Madras**

my classes notes with Prof. Anurag (**Deep Learning)**

Acknowledgments & Credits are mentioned to inspirational resources presented in the end of presentation. For more like this, https://sites.google.com/view/AIforEveryone

Want a Book?



Francois Chollet

Artificial Intelligence Researcher,
Google

Check free chapters at Manning Website!

Deep Learning with Python
https://www.manning.com/books/deep-learning-with-python

ISBN-13: 978-1617294433
Book Published: December 22, 2017

Acknowledgments & Credits are mentioned to inspirational resources presented in the end of presentation. For more like this, https://sites.google.com/view/AIforEveryone