Neural Network

Dr. Kelly Trinh

Housekeeping

- Collaborate session. Please check Collaborate Schedule document

WEEK	TECHNICAL COLLABORATE (RUN BY DR. KELLY TRINH)	PRACTICAL COLLABORATE (RUN BY DR. MARTHA COOPER)
WEEK 1: 3/5-9/5 WEDNESDAY & THURSDAY (6-7PM AEST)	Neural Network	NN implementation using Python
WEEK 2: 10/5-15/6 WEDNESDAY & THURSDAY (6-7PM AEST)	Regularisation, Optimisation and practical issue	AWS SageMaker, NN implementation on AWS
WEEK 3: 17/5-23/5 **THURSDAY** (6-7PM AEST)	**No technical collaborate session**	Hyperparameter Optimisation on AWS
WEEK 4: 24/5-30/5 WEDNESDAY & THURSDAY (6-7PM AEST)	Convolution Neural Network (CNN)	Implementation of CNN
WEEK 5: 31/5-6/6 **MONDAY** & THURSDAY (6-7PM AEST)	Advanced CNN and object detection	Implementation of CNN
WEEK 6: 7/6-13/6 WEDNESDAY (6-7PM)	QA session (Review Assessment 2, Students discuss data used in Assessment 3)	**No practical collaborate session**

Housekeeping

- Assessments:

Assessment task 1 [Video presentation] [15 %]	Released on Monday, Week 0. Due on Sunday 11:59pm (AEST) Week 2
Assessment task 2 [Report] [40 %]	Released on Monday, Week 2. Due on Sunday 11:59pm (AEST) Week 4
Assessment task 3 [Report] [45 %]	Released on Monday, Week 2. Due on Wednesday 11:59pm(AEST) Week 7

- Please read **section 4.1** of the course outline for special consideration, extension request and late submission.
- All special consideration and extension requests are submitted via JCU website `https://www.jcu.edu.au/students/assessment-and-results/special-consideration`

AWS reimbursement

- Please set up free tier AWS account.
- College of Science and Engineering will reimburse **up to $40** per student for AWS service.
- Note that if you exceed the amount, the exceeded amount will not be approved for an reimbursement.
- Please check **AWS Discussion Board** regarding how to apply for the reimbursement.

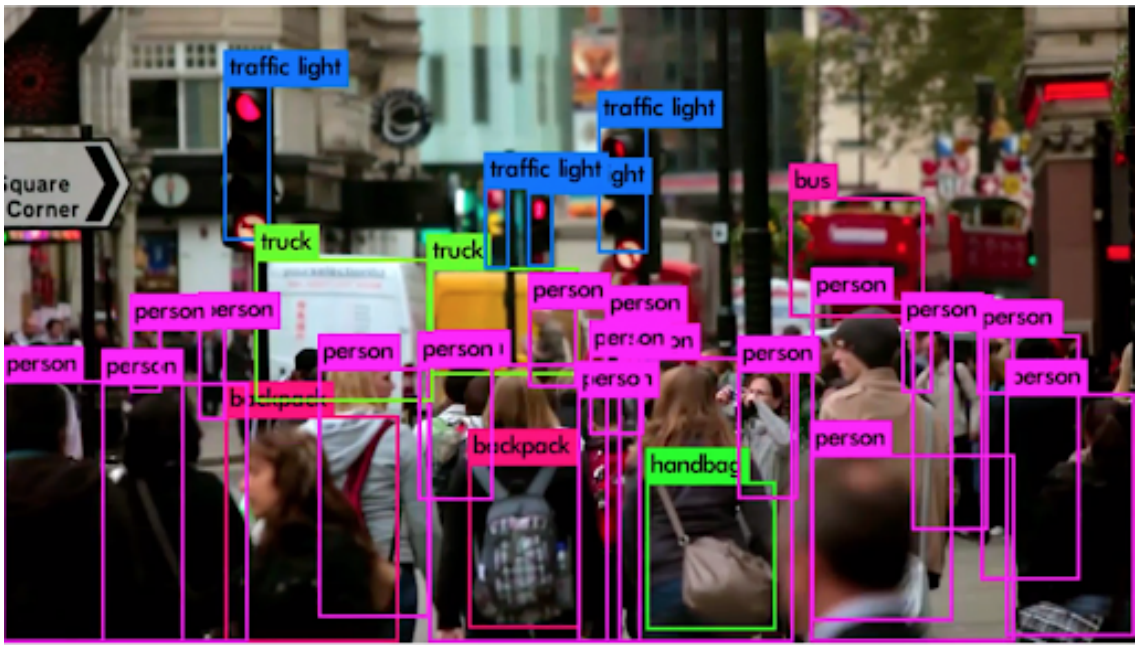
Recommended Activities for Week 1 and 2

- Install Python3, Jupyter notebook, Tensorflow 2 and Keras
- Set up an AWS free tier account
- Read Chapter 6, 7, 8, 9 and 16: Machine Learning in the AWS Cloud by Mishra, Abhishek (2019) as well as online content of Week 2 and 3

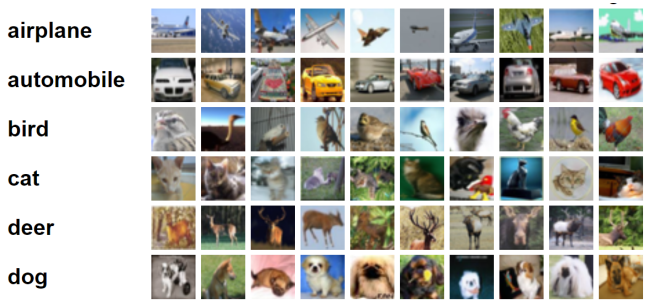
Neural Network

Some applications of neural networks

- Neural network is a rapidly grown research area. It has been used for classification, prediction and diagnosis in many research areas such as health, finance, agriculture, IoT, and etc.
- In this subject, we will focus on
  - Vanilla neural network (Multilayer perceptron Neural Network)
  - Convolution neural network (Image classification, Image detection)
  - Implementation of the neural network on AWS



Source: Medium

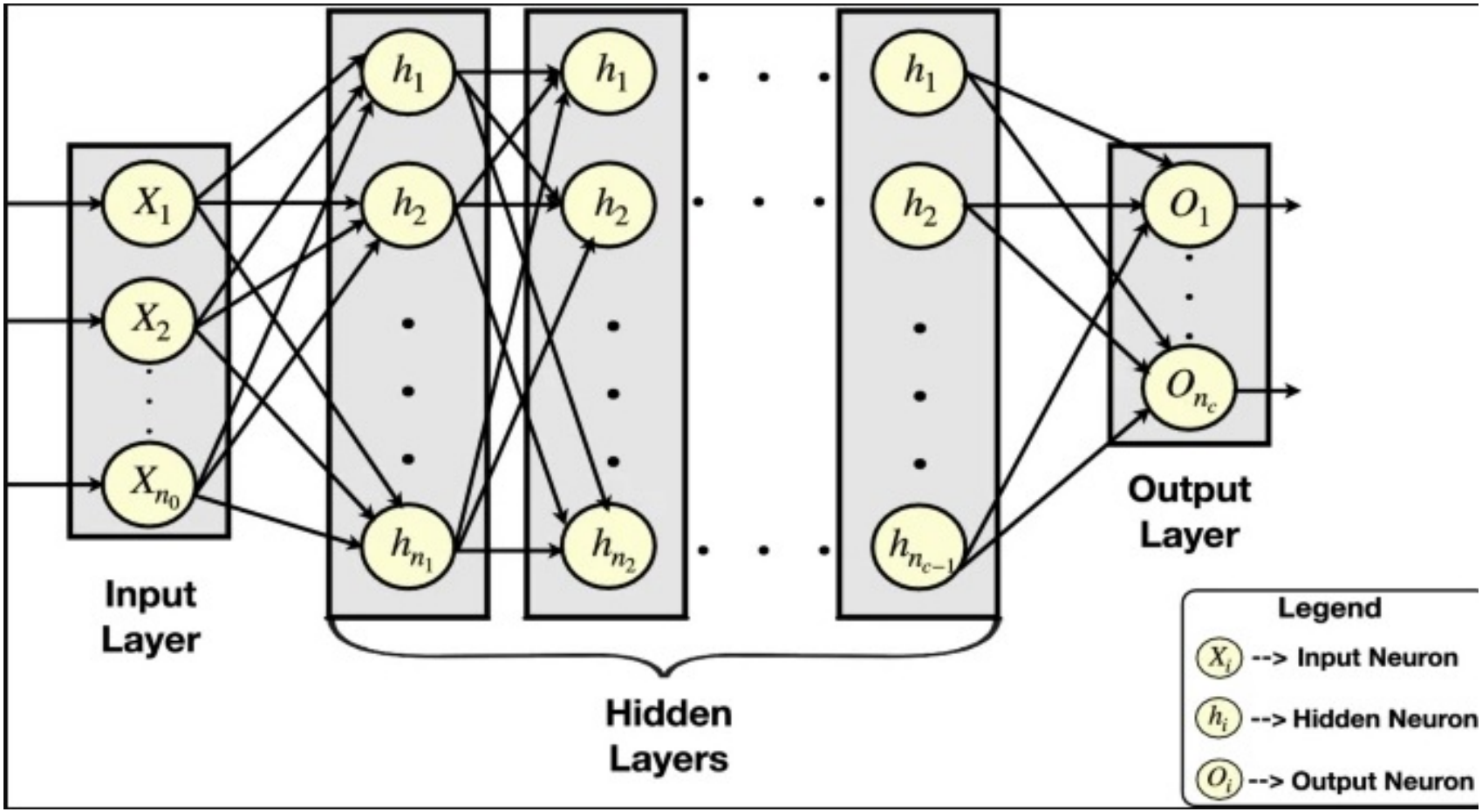


Multilayer perceptron Neural network

How can neural network learn patterns of data, and provide highly accurate classification, and prediction?

- Structure of neural network
  - Neuron (perceptron), activation function
  - Input, output, and hidden layers
- How is a neural network estimated?

Structure of neural network

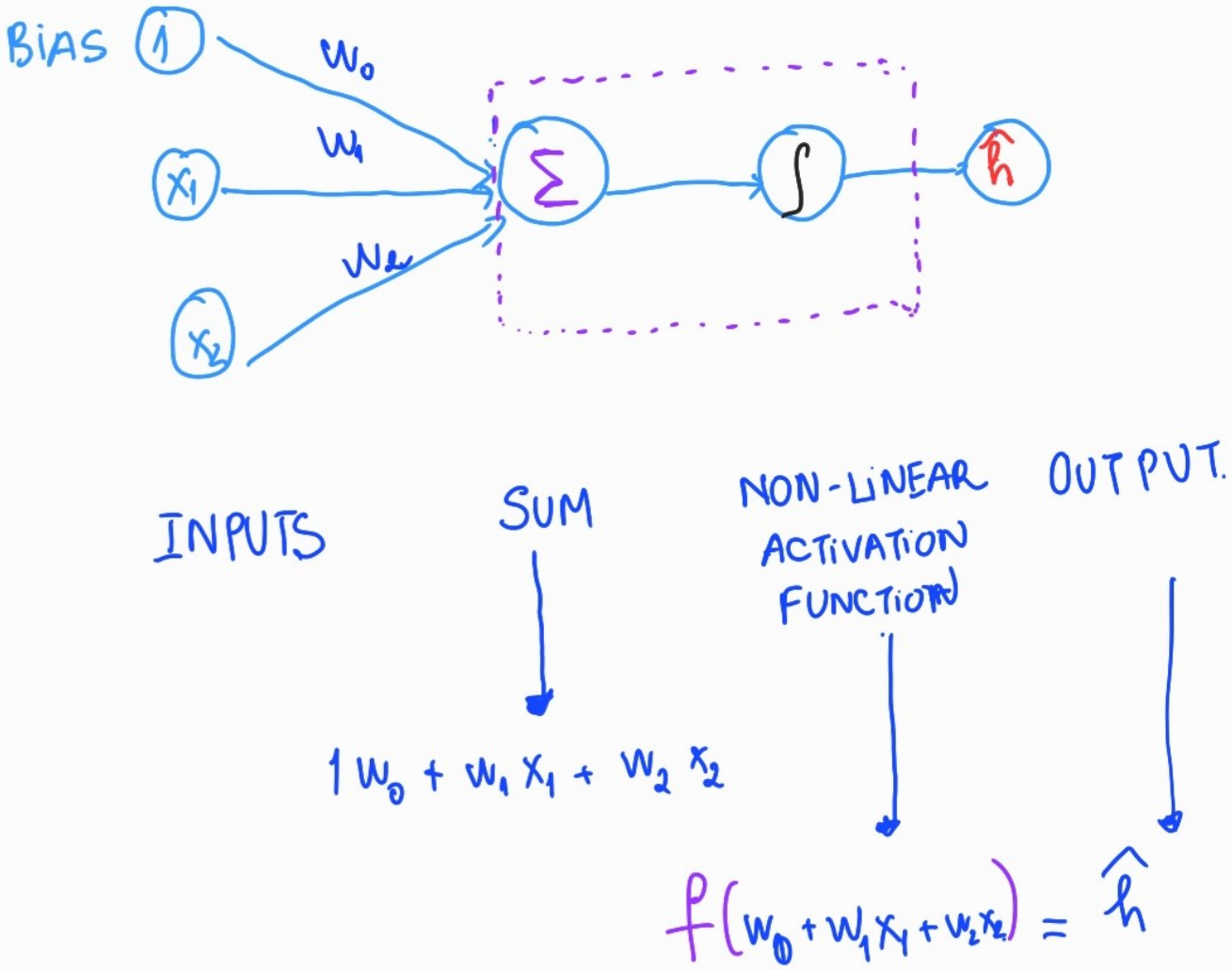


Source: Rodenas et.al (2021)

What is a neuron and how is it constructed?

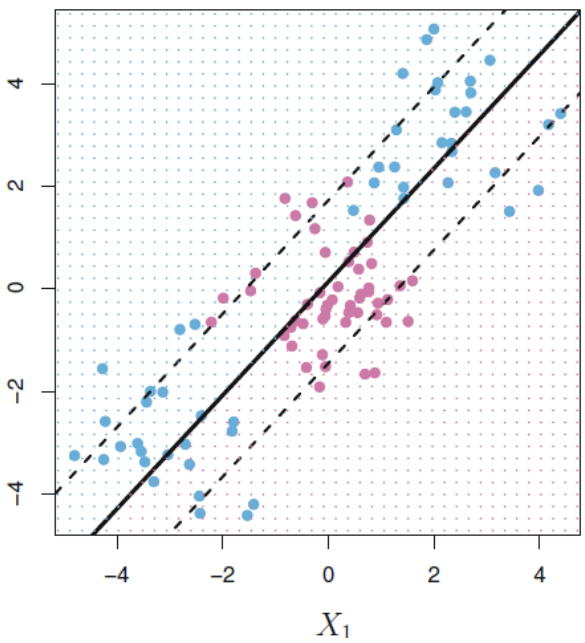
How is a neuron constructed?

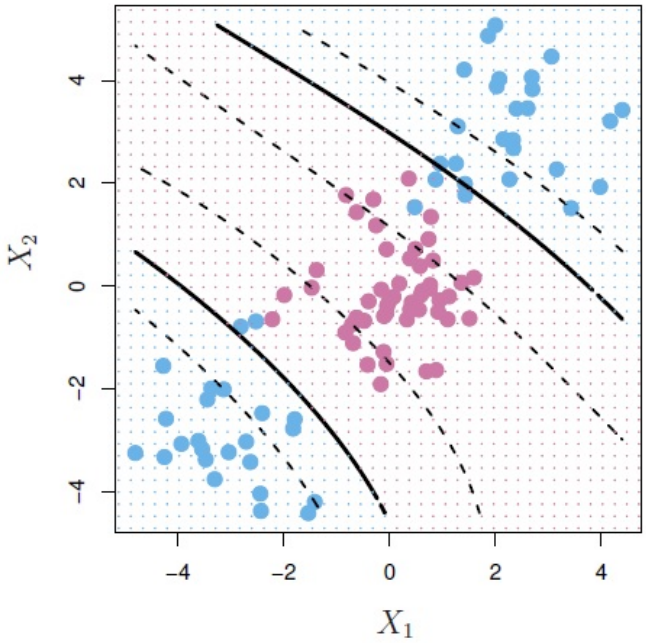
Neuron is simply a number carrying information of inputs.

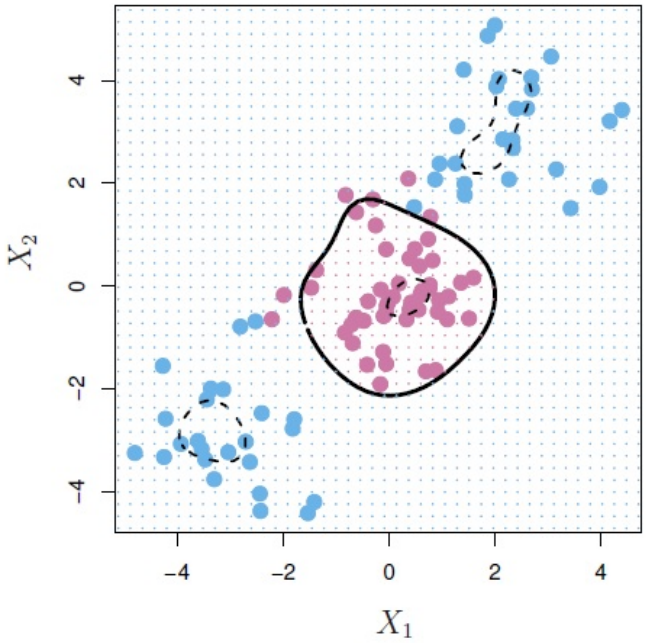


What are activation functions?

Activation functions introduces non-linearity in neural networks. They allow neural network to better approximate complex relationship between data.

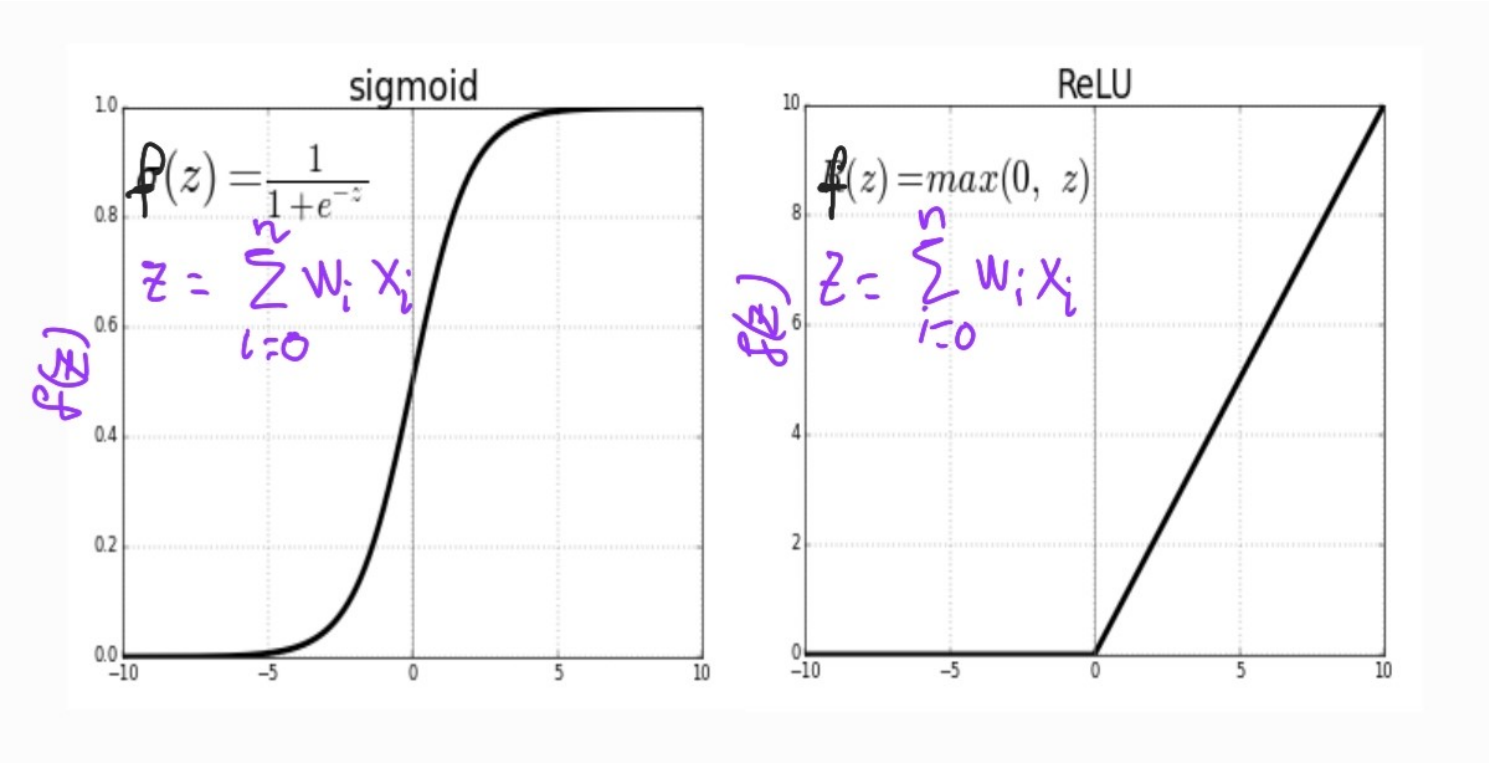






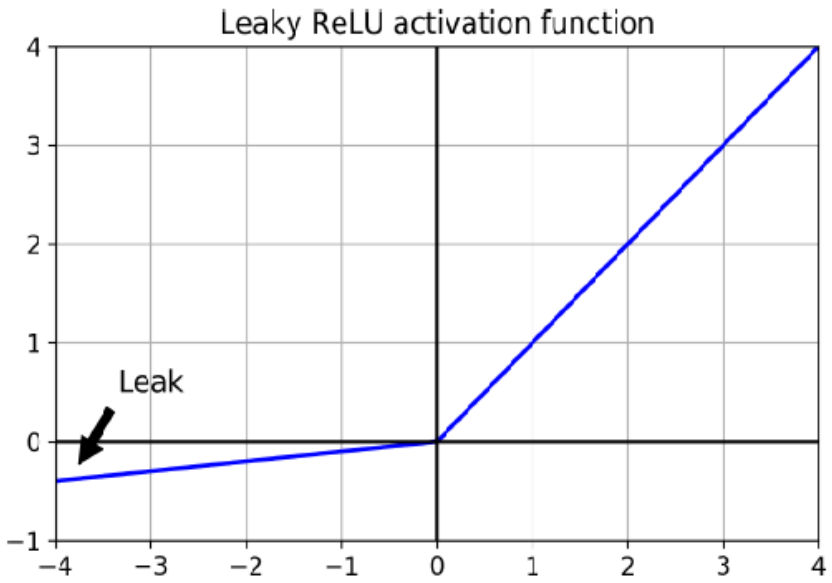
Source:ISLR

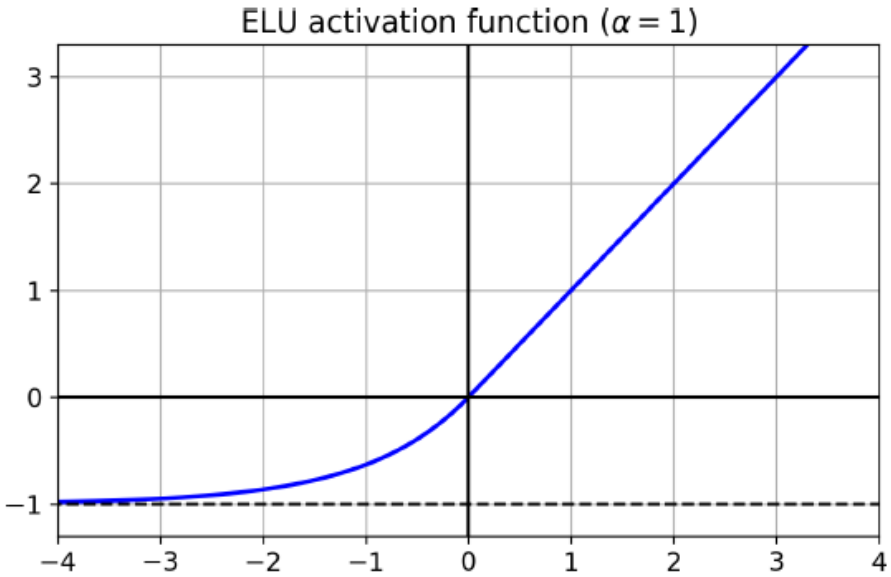
Activation functions



Source: Towards Data Science

Activation function

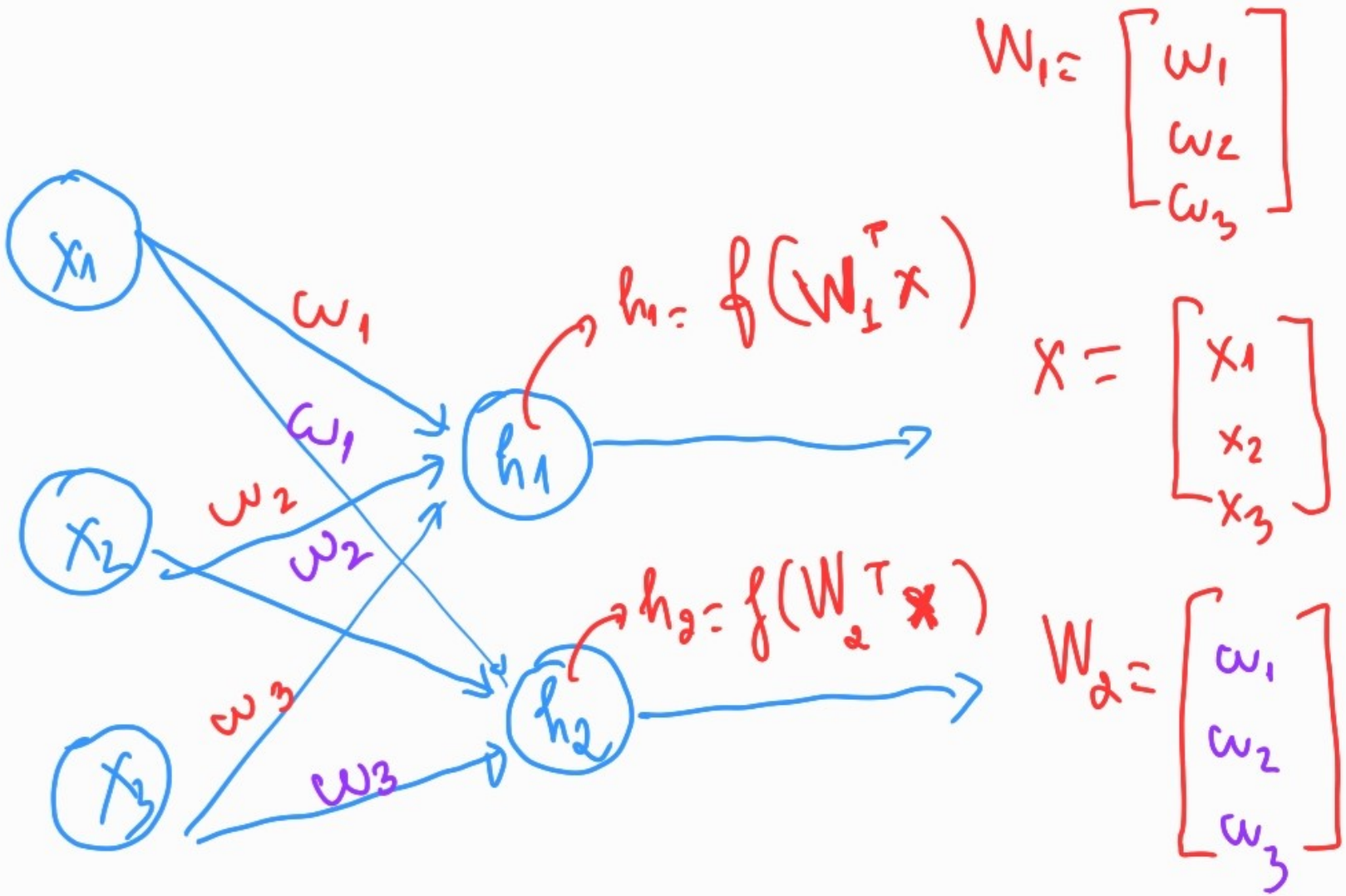




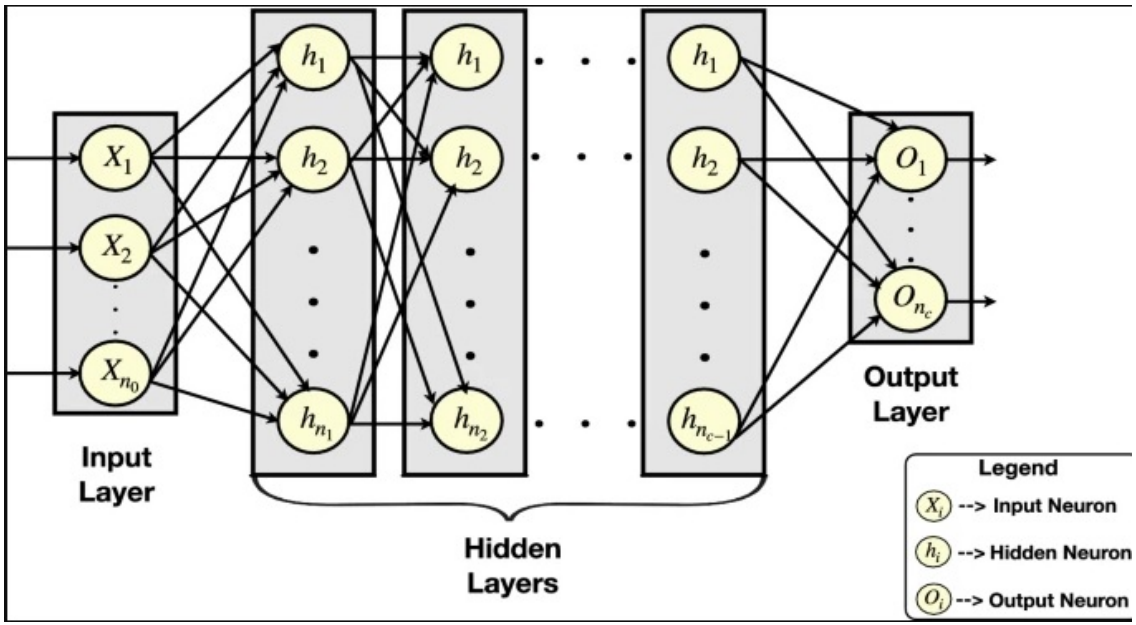
$$ELU_\alpha(z) = \begin{cases} \alpha(\exp(z) - 1) & \text{if } z < 0 \\ z & \text{if } z \geq 0 \end{cases}$$

- ELU (exponential linear unit) has a slower computation than ReLu and Leaky ReLu

Multiple neurons



Structure of Neural Network



Source: Rodenas et.al (2021)

All inputs are connected to each neuron, these layers are called **Dense** layers

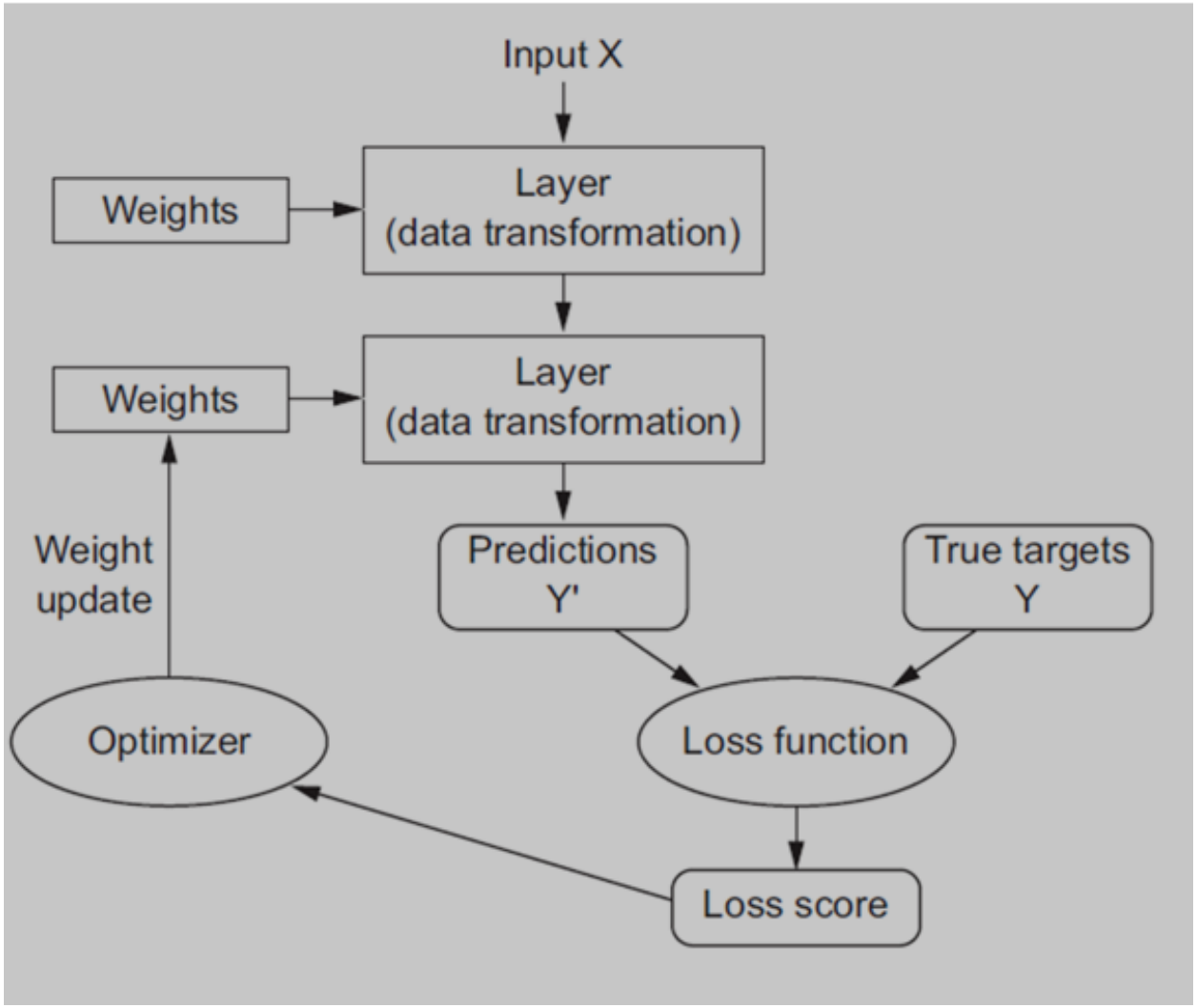
- Python code

```
model = tf.keras.models.Sequential([
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(64, activation=tf.nn.relu),
        tf.keras.layers.Dense(32, activation=tf.nn.relu),
        tf.keras.layers.Dense(10, activation=tf.nn.softmax)
    ])
```

How are the weights and bias in neurons estimated?

- Loss function
- Optimiser
  - Gradient
  - Chain rule

How a neural network is estimated?



How a neural network is estimated?

- Aim to minimise the loss incurred from incorrect predictions/classification
- The optimisation object is defined as

$$J(\mathbf{W}) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(f(x_i;\mathbf{W}), y_i)$$

where the loss function $\mathcal{L}(f(x_i;\mathbf{W}), y_i)$ can have the form as

- Mean squared error $\mathcal{L}(f(x_i;\mathbf{W}), y_i) = (y_i - f(x_i;\mathbf{W}))^2$
- Binary Cross entropy $\mathcal{L}(f(x_i;\mathbf{W}), y_i) = y_i log(f(x_i;\mathbf{W})) + (1 - y_i) log(1 - f(x_i;\mathbf{W}))$

Problem type	Last-layer activation	Loss function
Binary classification	`sigmoid`	`binary_crossentropy`
Multiclass, single-label classification	`softmax`	`categorical_crossentropy`
Multiclass, multilabel classification	`sigmoid`	`binary_crossentropy`
Regression to arbitrary values	None	`mse`
Regression to values between 0 and 1	`sigmoid`	`mse` or `binary_crossentropy`

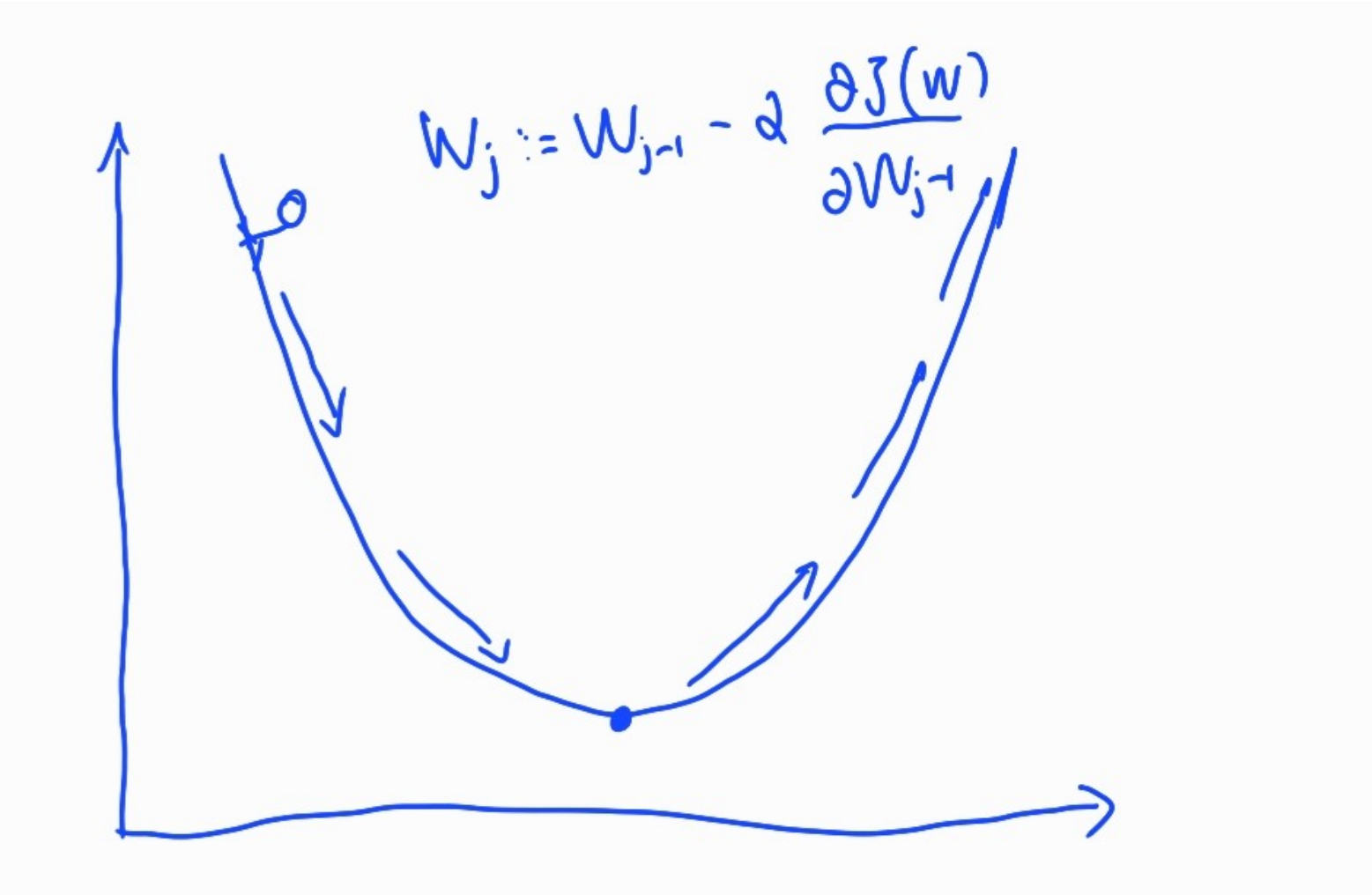
Optimiser

- Optimiser
  - Gradient
  - Chain rule

Optimisation algorithm - Gradient descent

- Initialize weights randomly
- Repeat until convergence $\{ \mathbf{W}_j := \mathbf{W}_{j-1} - \alpha \frac{\partial J(\mathbf{W})}{\partial \mathbf{W}_{j-1}} \}$
- Return weights

$\frac{\partial J(\mathbf{W})}{\partial \mathbf{W}_{j-1}}$ tell us how much the loss $J(\mathbf{W})$ change due to a small change in $\mathbf{W}$



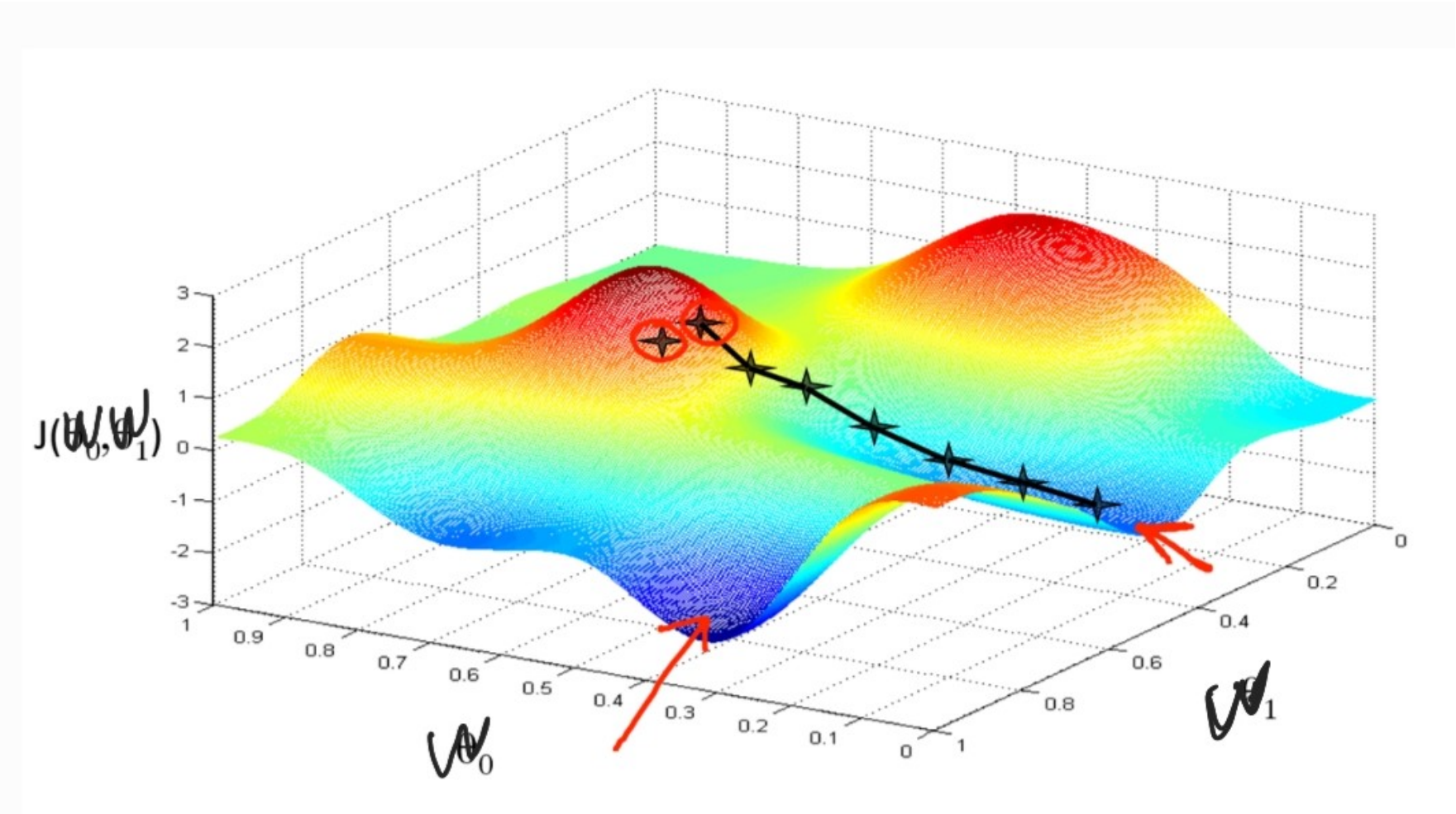
How to compute the gradient?

Chain rule

$$\frac{\partial J(w)}{\partial w_2} = \frac{\partial J(w)}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial w_2}$$

$$\frac{\partial J(w)}{\partial w_1} = \frac{\partial J(w)}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial h_1} \cdot \frac{\partial h_1}{\partial w_2}$$

Estimation of neural network



From: Medium website

Neural Network in Python

- Train and Deploy NN model

```
model.compile(loss=categorical_crossentropy,
              optimizer=SGD(lr=0.001, decay=1e-6, momentum=0.9, nesterov=True),
              metrics=['accuracy',
                       tf.keras.metrics.Precision(),
                       tf.keras.metrics.Recall()])
model.fit(x_train, y_train, batch_size=10,
          validation_split=0.1,
          epochs=2)
print(model.summary())
```

The lab session

- Implementation of a neural network using Tensorflow2.0 and Python3
- Next week: Regularisation and practical issues relating to Neural network

Reading list

- Chapter 10 and 11 in "Hands-on MAchine Learning with Scikit-Learn Keras and Tensorflow" Aurelien Geron (2019)