# 00 - INTRODUCTION

# Contents

Artificial Intelligence vs Machine Learning vs Deep Learning

Types of Learning

- *Supervised*
- *Unsupervised*
- *Reinforcement*

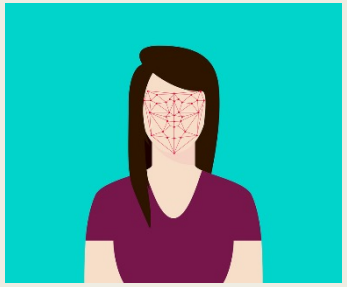
# MACHINE LEARNING vs DEEP LEARNING vs ARTIFICIAL INTELLIGENCE

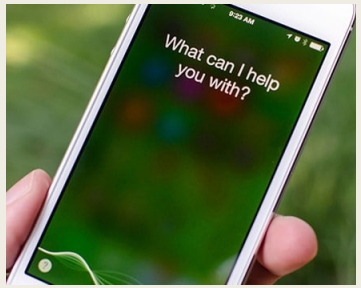
# Introduction

Artificial Intelligence/Machine learning does not only mean robots or Sci-Fi movies!

Machine and deep learning applications are everywhere!

Google search engine, amazon recommender systems, Facebook facial recognition (tagging), Siri





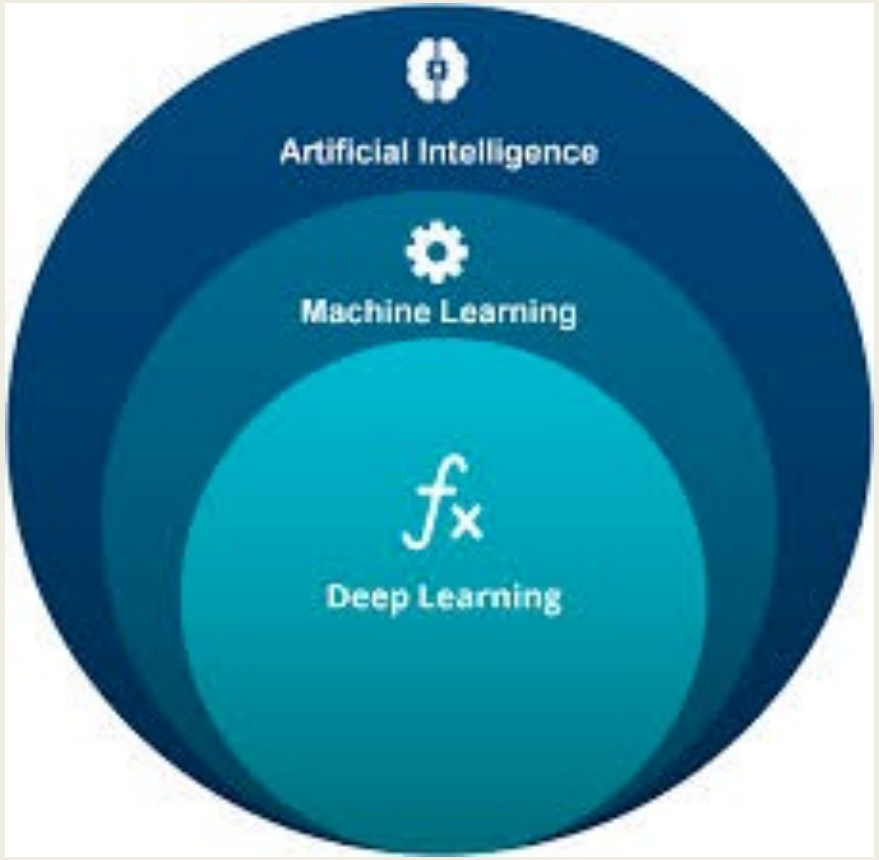


amazon


# Big Picture - AI

**Artificial Intelligence**

Science that empowers computers to mimic human intelligence such as decision making, text processing, and visual perception.



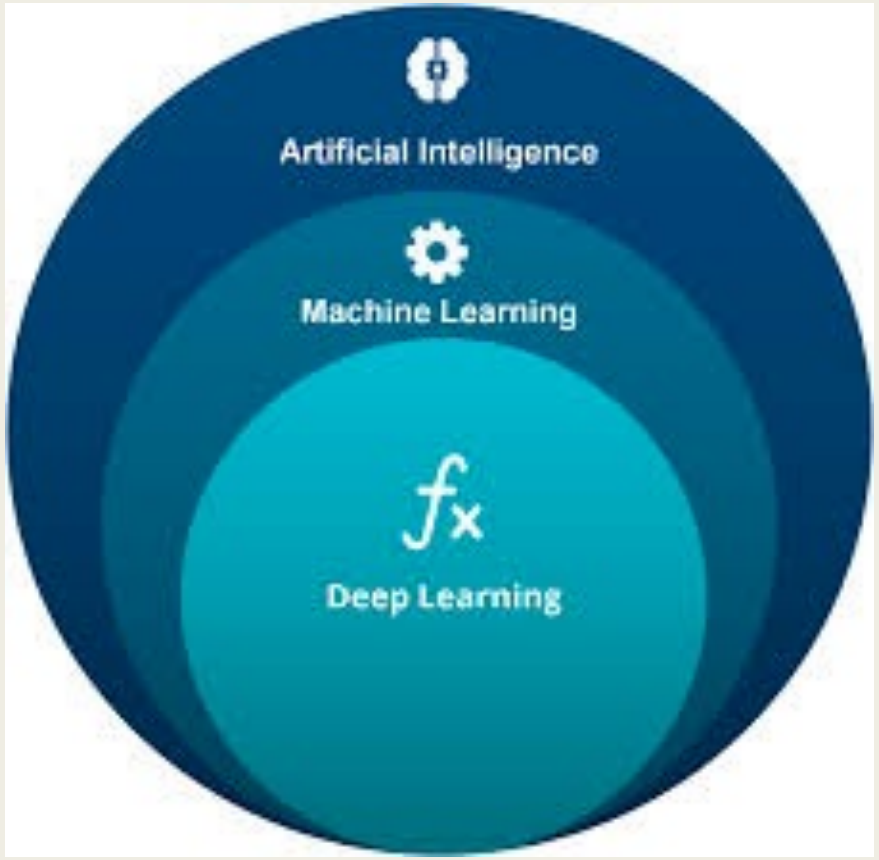
# Big Picture - ML

**Machine Learning**

A subfield of Artificial Intelligence that enables machines to improve at a given task with experience.

All machine learning techniques are classified as Artificial Intelligence but not all Artificial Intelligence could count as Machine Learning.

e.g. some basic Rule-based engines could be classified as AI but they do not learn from experience therefore they do not belong to the machine learning category.



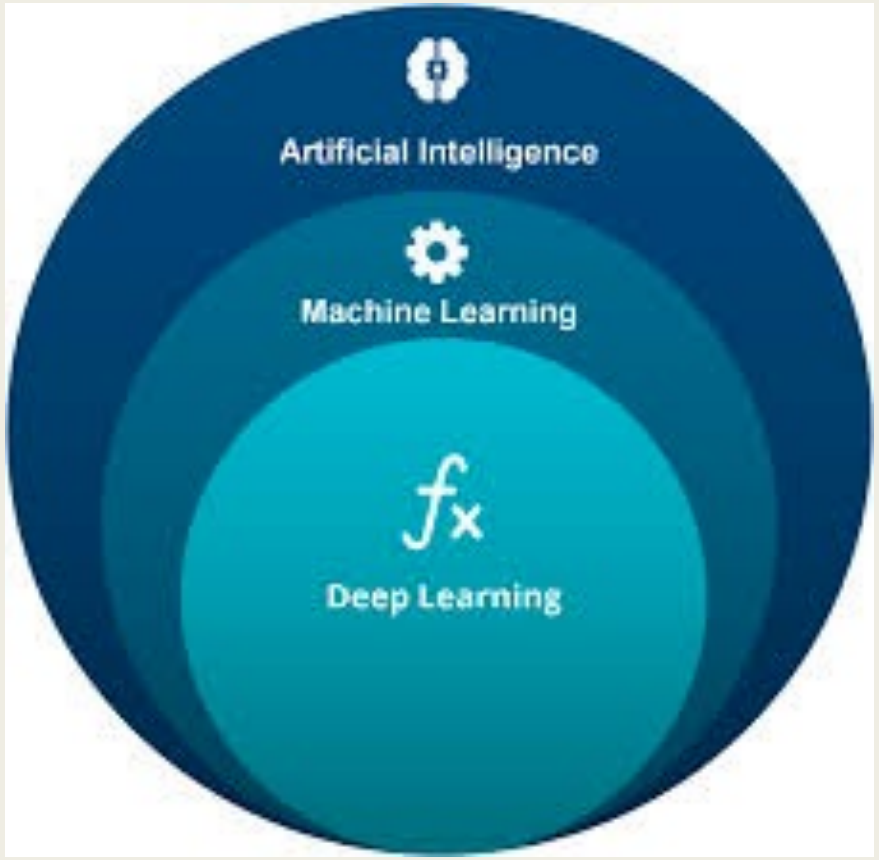
# Big Picture - DL

**Deep Learning**

A specialized field of Machine Learning that relies on training of Deep Artificial Neural Networks (ANNs) using large dataset such as images.

ANNs are information processing models inspired by the human brain.

The human brain consists of billions of neurons that communicate to each other using electrical and chemical signals and enable humans to see, feel, and make decision.

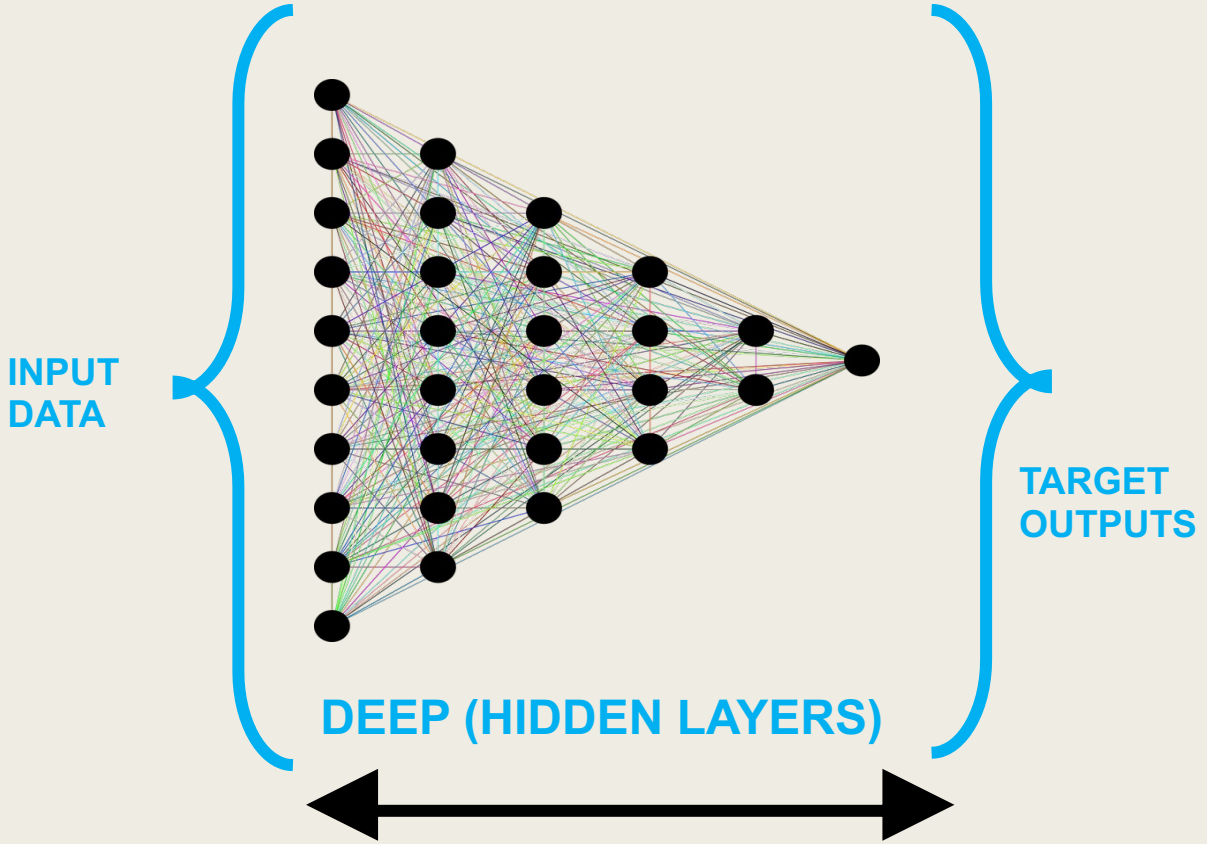


# Big Picture - DL

**Deep Learning**

Depth is a measure of how many hidden layers

More than three layers (including input and output) qualifies as“**deep**” **learning.**



# AI vs ML vs DL

Deep learning can automatically extract features automatically:

**Machine learning Process:**

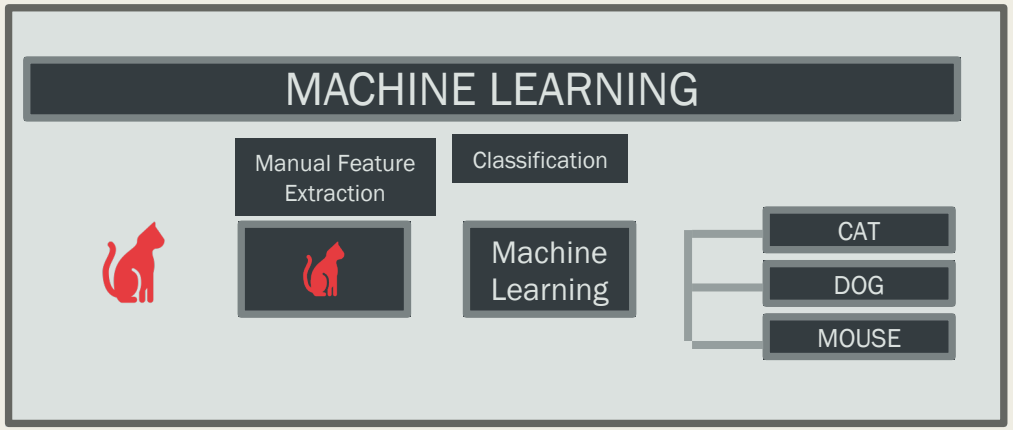
(1) select the model to train,

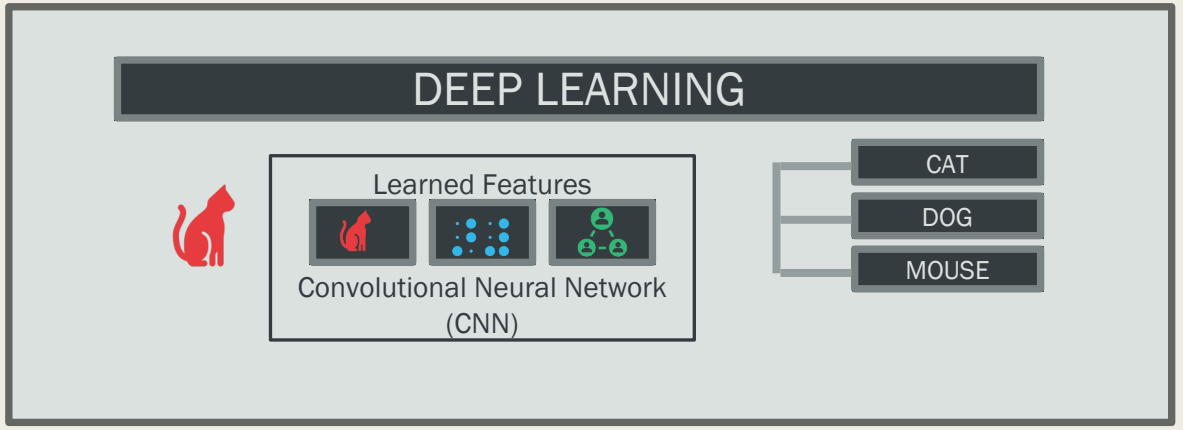
(2) manually perform feature extraction.

**Deep Learning Process:**

(1) select the architecture of the network,

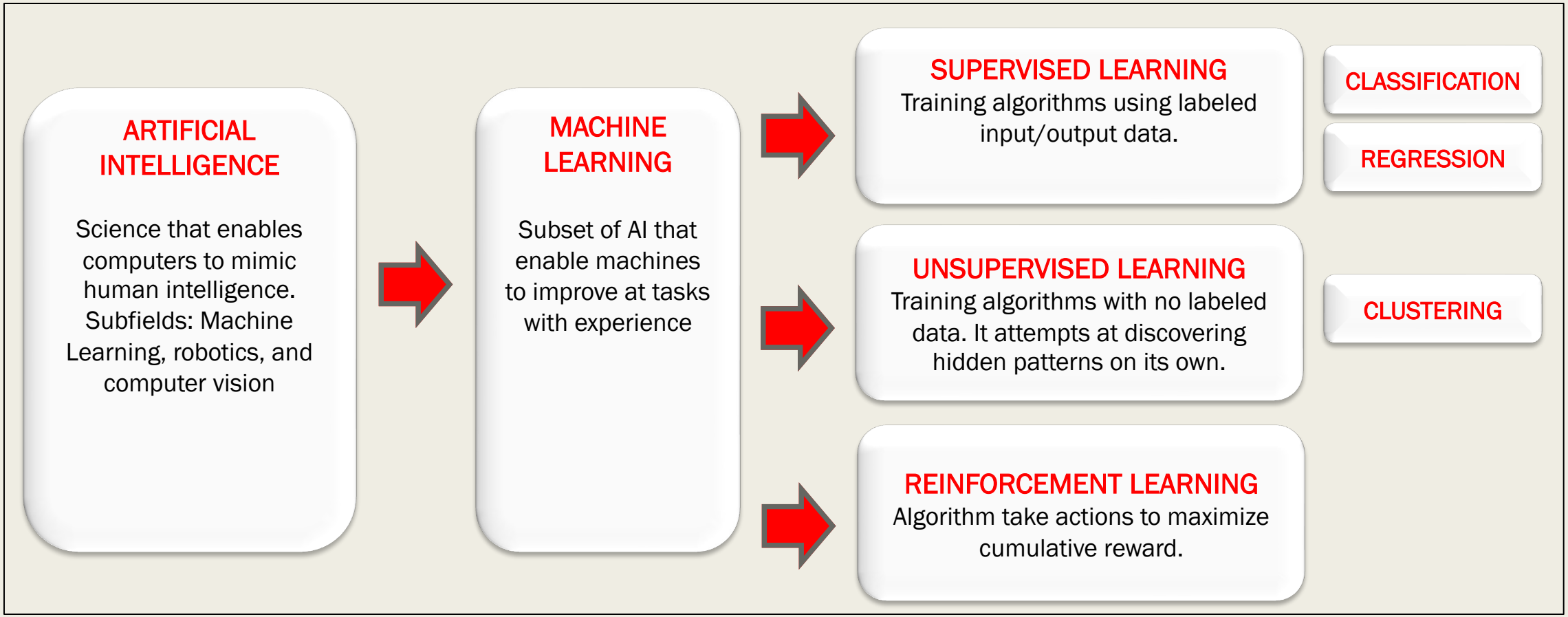
(2) features are automatically extracted by feeding in the training data (such as images) along with the target class (label).



# TYPES OF LEARNING

# Types of Learning

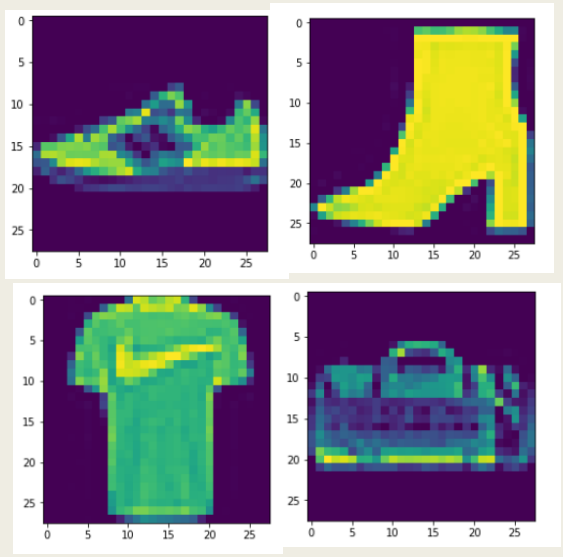


# Supervised Learning

**Supervised:** used to train algorithms using labeled input and output data.

Performance is assessed by comparing trained model prediction vs. real output.

**INPUT DATA** $(X\_train)$



MODEL

**DESIRED OUTPUT TARGET CLASSES / LABELS** $(y\_train)$

- T-SHIRT/TOP
- TROUSER
- PULLOVER
- DRESS
- COAT
- SANDAL
- SHIRT
- SNEAKER
- BAG
- ANKLE BOOT

Fashion Data Set

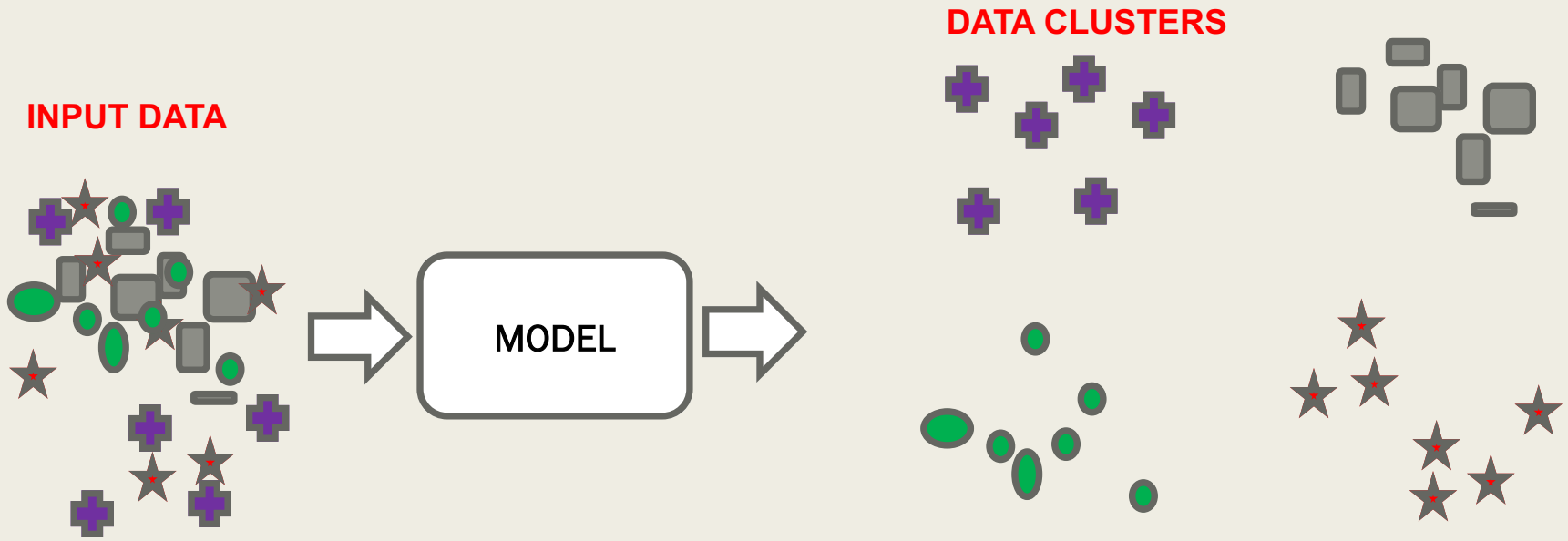
# Unsupervised Learning

**Unsupervised learning:** provides the algorithm with no labeled data.

The algorithm attempts at discovering hidden patterns within the training data.

Unsupervised learning methods can analyze complex data that humans might find difficult to interpret.

No feedback!



# Reinforcement Learning

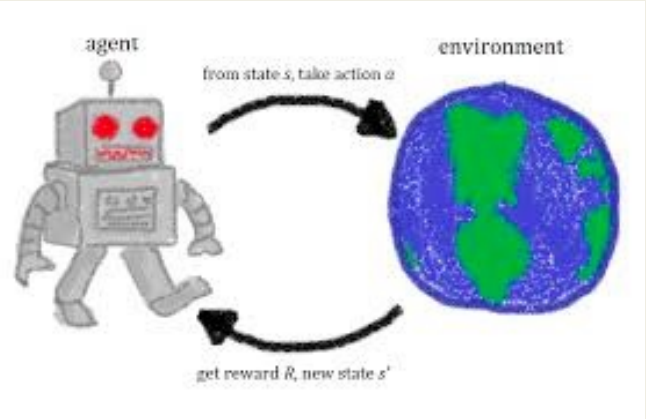
Reinforcement learning allows machines take actions to maximize cumulative reward.

Reinforcement algorithms learn by trial and error through reward and penalty.

Two elements: **environment** and **learning agent**.

The environment rewards the agent for correct actions.

Based on the reward or penalty, agent improves its environment knowledge to make better decision.


**Key Words**

Trial
&
Error

# 01 - REGRESSION

# Contents

# CAR SALES PROBLEM STATEMENT

# Problem Statement

You are working as a car salesman and you would like to develop a model to predict the total dollar amount that customers are willing to pay given the following attributes:

- *Customer Name*
- *Customer e-mail*
- *Country*
- *Gender*
- *Age*
- *Annual Salary*
- *Credit Card Debt*
- *Net Worth*

**This is a REGRESSION TASK**

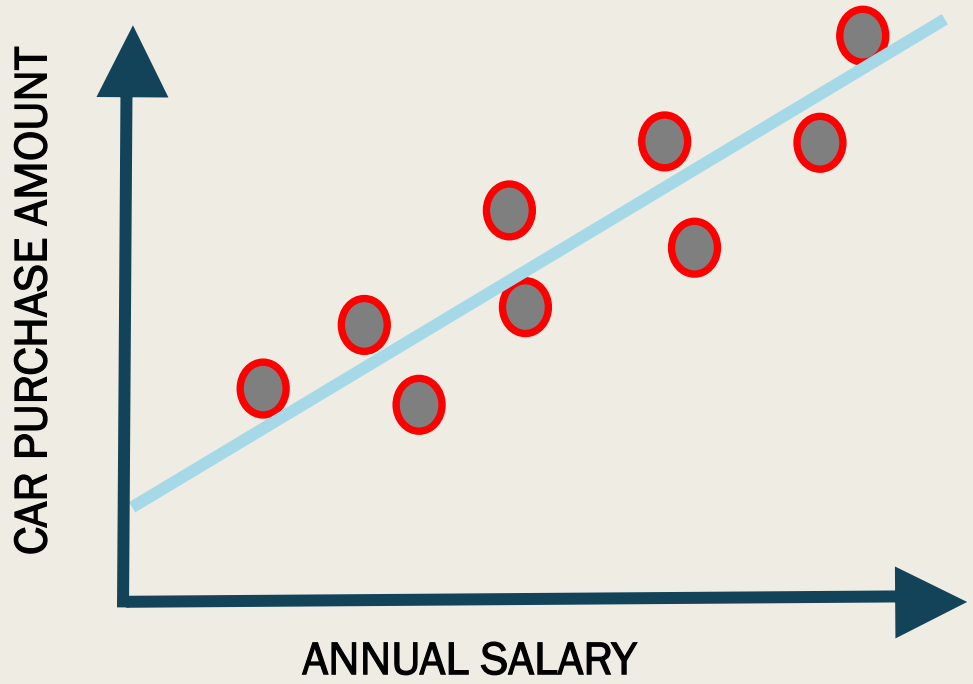
The model should predict:

- *Car Purchase Amount*

# Regression

Regression works by predicting value of one variable **y** based on another variable **X**.

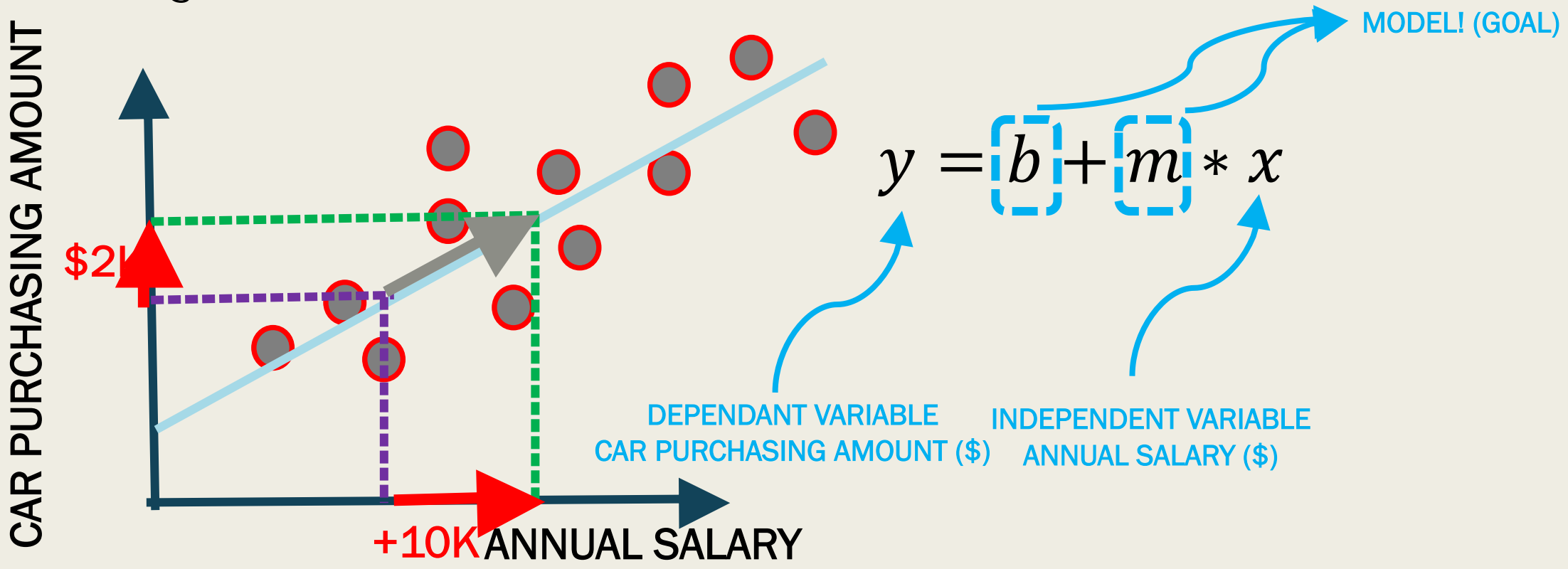
**X** is called the ***independent variable*** and **y** is called the ***dependant variable***.




| Annual Salary | Car Purchase Amount |
|---|---|
| 62812 | 35321 |
| 66647 | 45116 |
| 53799 | 42926 |
| 79370 | 67422 |
| 59729 | 55915 |
| 68500 | 56612 |
| 39815 | 28926 |
| 51752 | 47435 |
| 58139 | 48014 |
| 53457 | 38190 |
| 73349 | 59046 |
| 55422 | 42289 |
| 37336 | 28700 |
| 68304 | 49259 |
| 72776 | 49510 |

# Regression

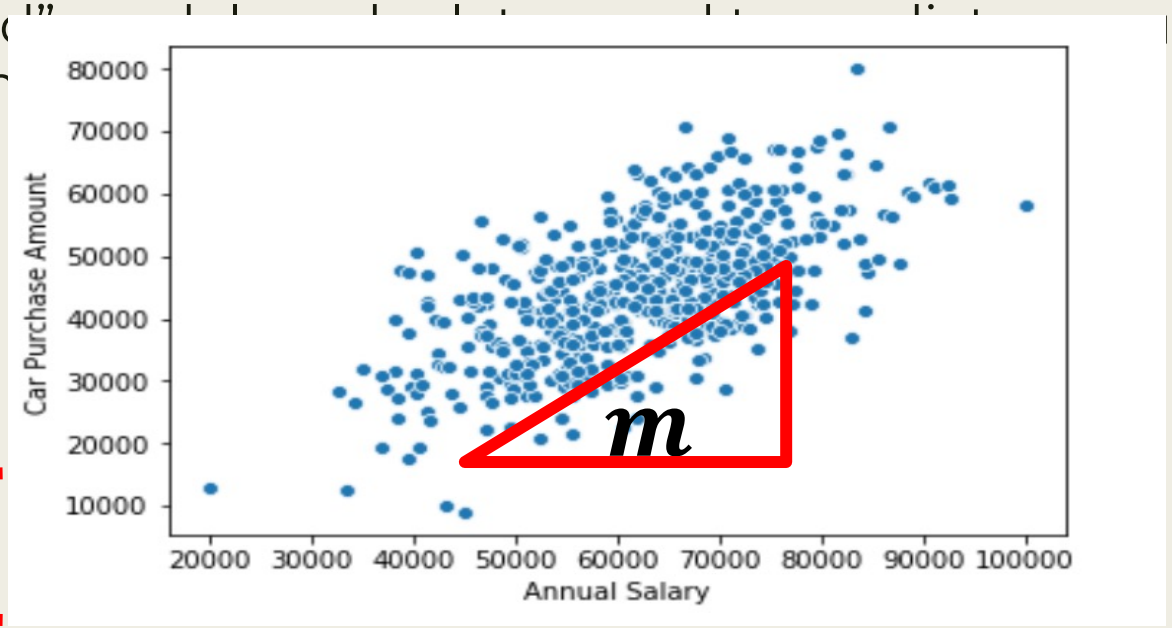
Goal is to obtain a relationship (model) between the Annual salary and car purchasing amount.



# Regression

Once the coefficients '**m**' and '**b**' are obtained, you have obtained a regression model!

This "trained" [illegible] rchase amount (dollars) based on th[illegible]




$Y$

$b$

$X$

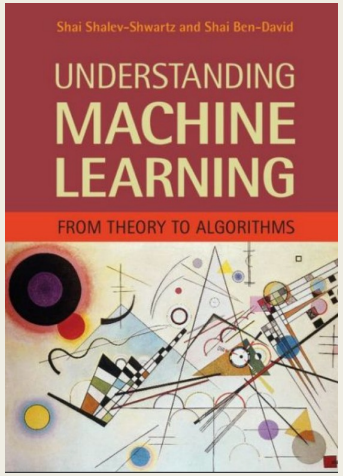
$$y = mX + b$$

DEPENDANT VARIABLE

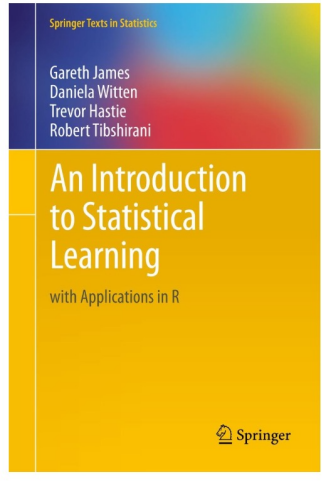
INDEPENDANT VARIABLE

# Additional Resources

Additional Resources, Page #123:
http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf




- Additional Resources, Page #61:
- http://www-bcf.usc.edu/~gareth/ISL/ISLR%20Seventh%20Printing.pdf



# ARTIFICAL NERURAL NETWORKS

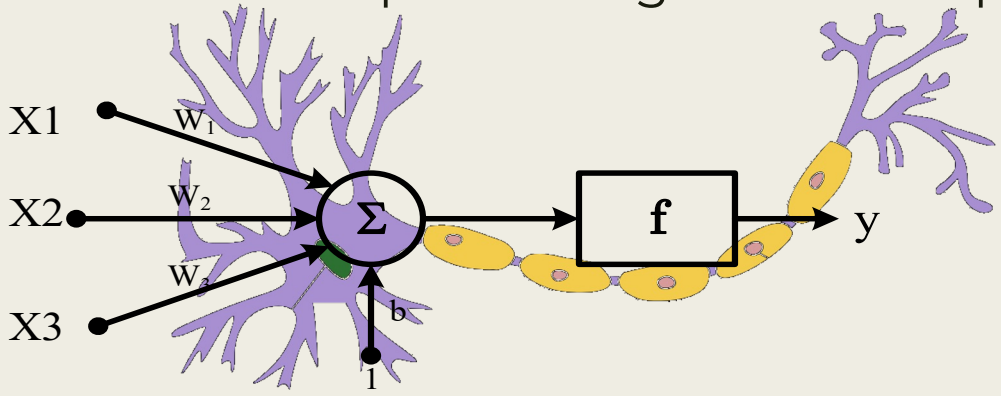
# A.N.N. - Introduction

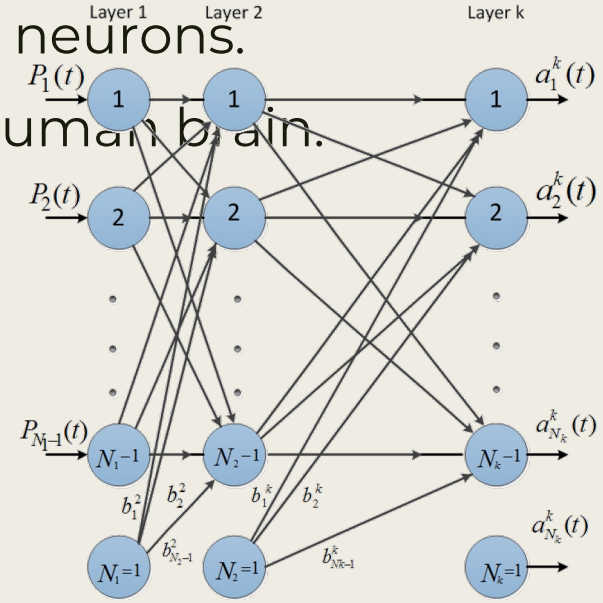
The brain has over 100 billion neurons communicating through electrical and chemical signals.

Neurons communicate with each other and help us see, think, and generate ideas.

Human brain learns by creating connections among these neurons.

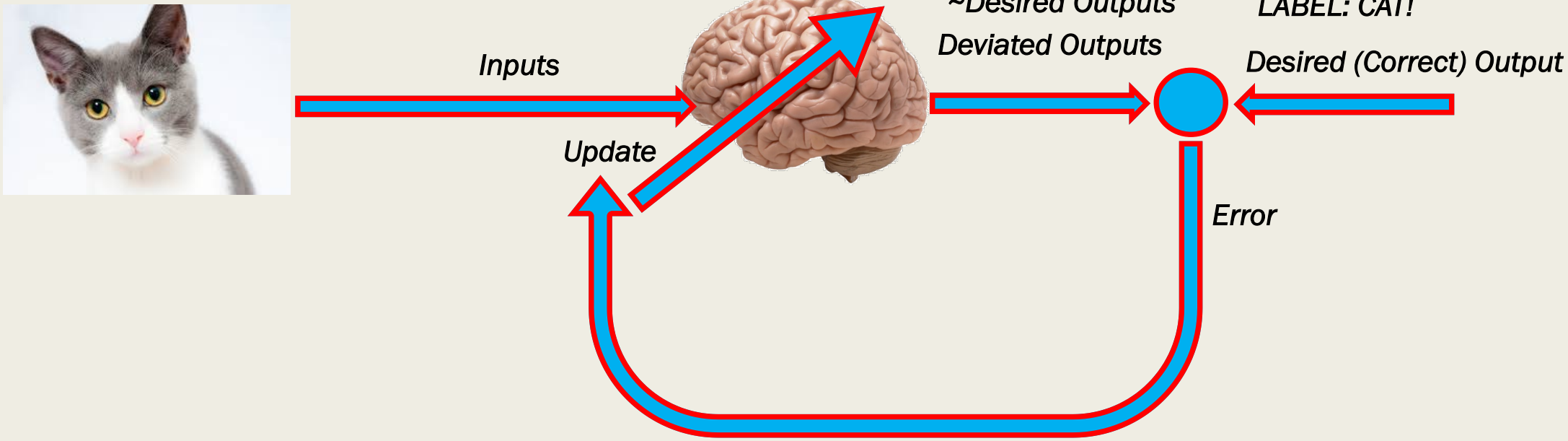
ANNs are information processing models inspired by the human brain.



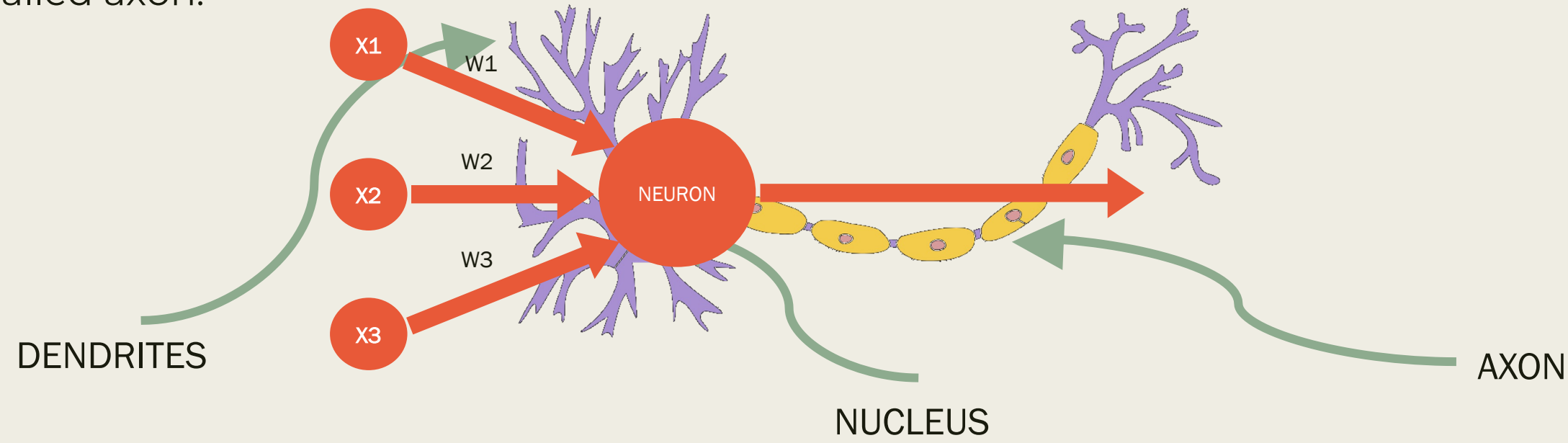
# How to humans learn ?

Humans learn from experience (by example)



# Neuron Mathematical Model

The neuron collects signals from input channels named dendrites, processes information in its nucleus, and then generates an output in a long thin branch called axon.

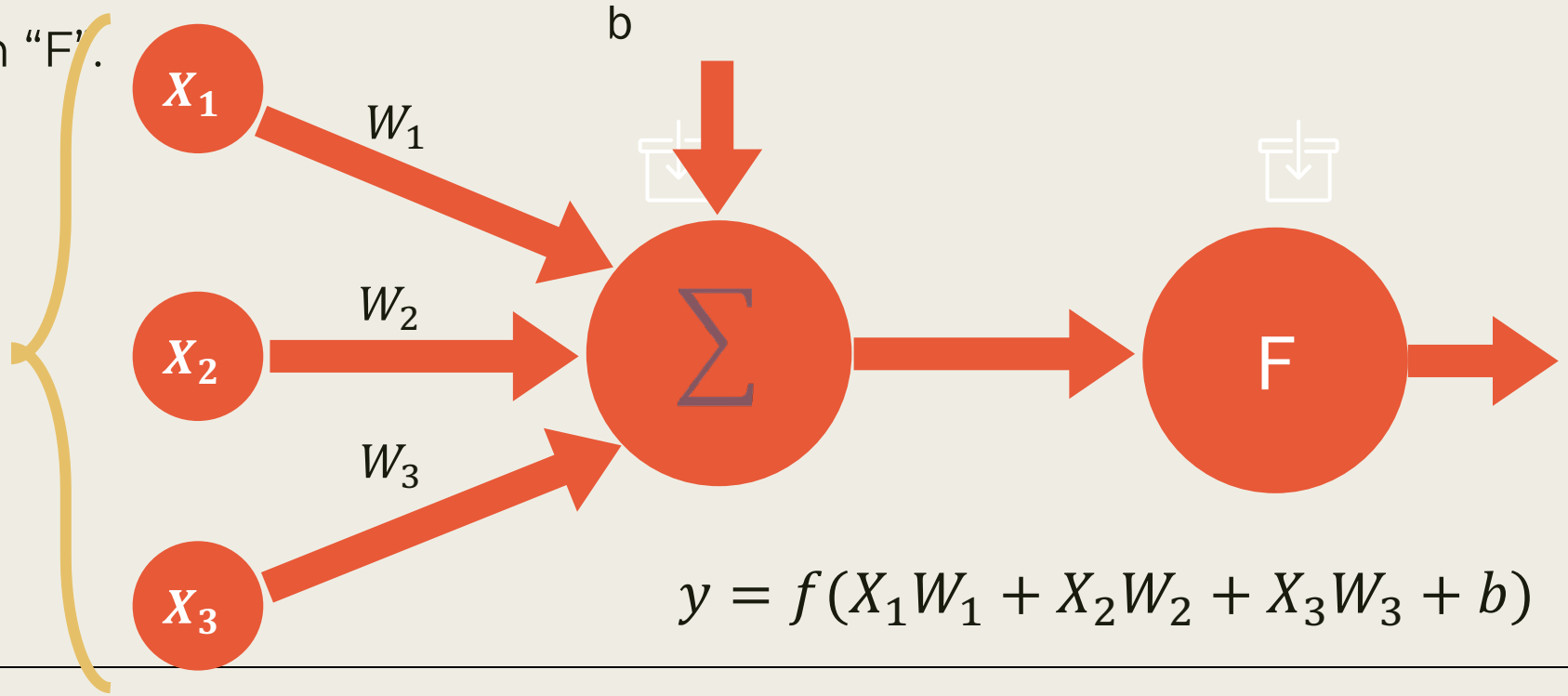


# Neuron Mathematical Model

Bias allows to shift the activation function curve up or down.

Number of adjustable parameters = 4 (3 weights and 1 bias).

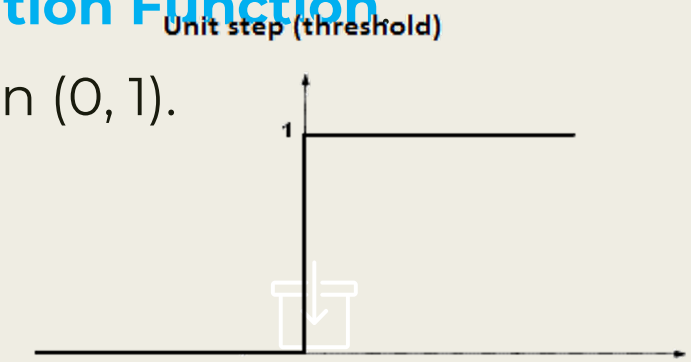
Activation function "F".



# Single Neuron Model

Assume that the activation function is a **Unit Step Activation Function**

The activation functions is used to map the input between (0, 1).







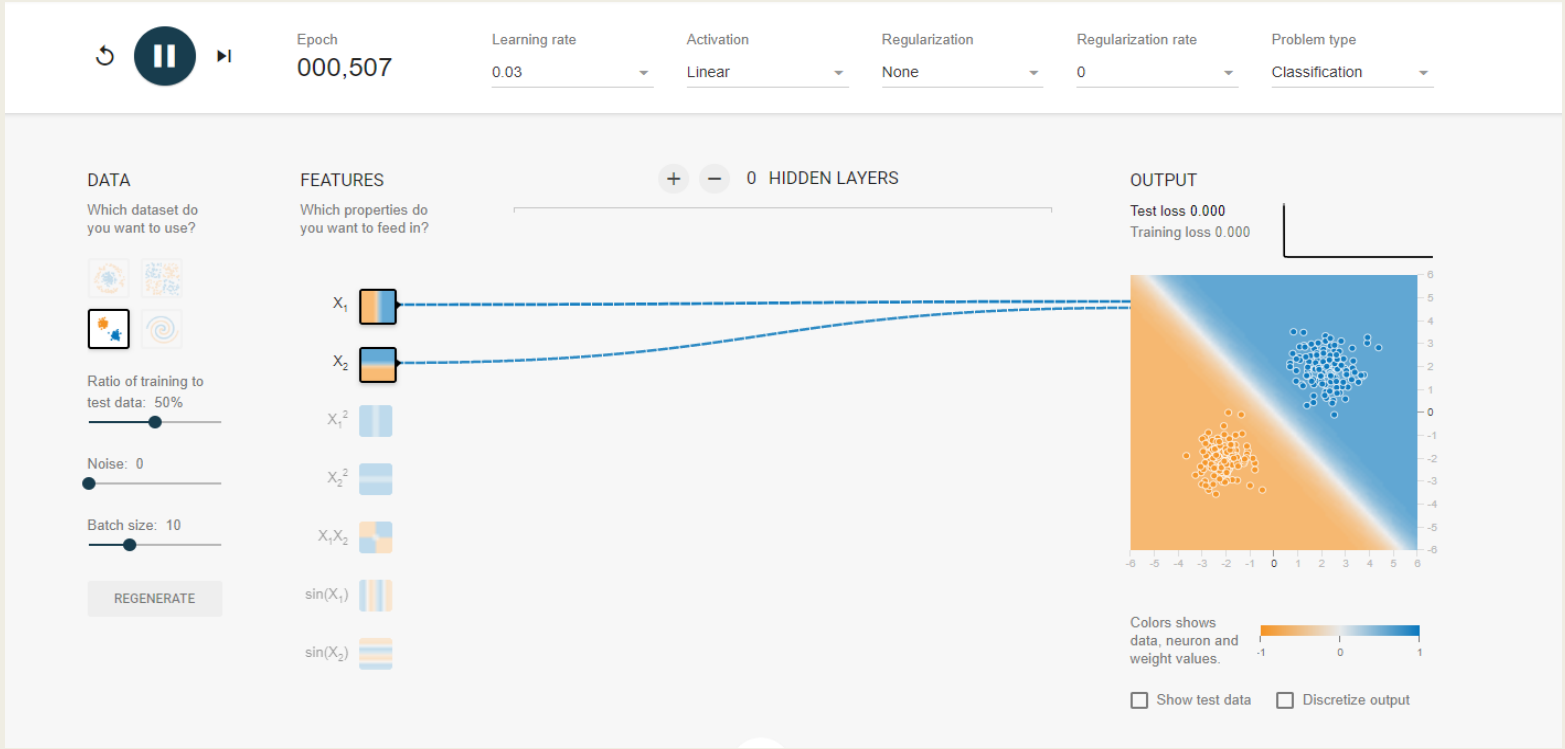
$$y = f(X_1W_1 + X_2W_2 + X_3W_3 + b)$$

$$y = f(1 * 0.7 + 3 * 0.1 + 4 * 0.3) = f(2.2)$$

$$y = 1 \text{ (because } 2.2 > 0)$$

# Single Neuron Model

Try a neural network out : https://playground.tensorflow.org

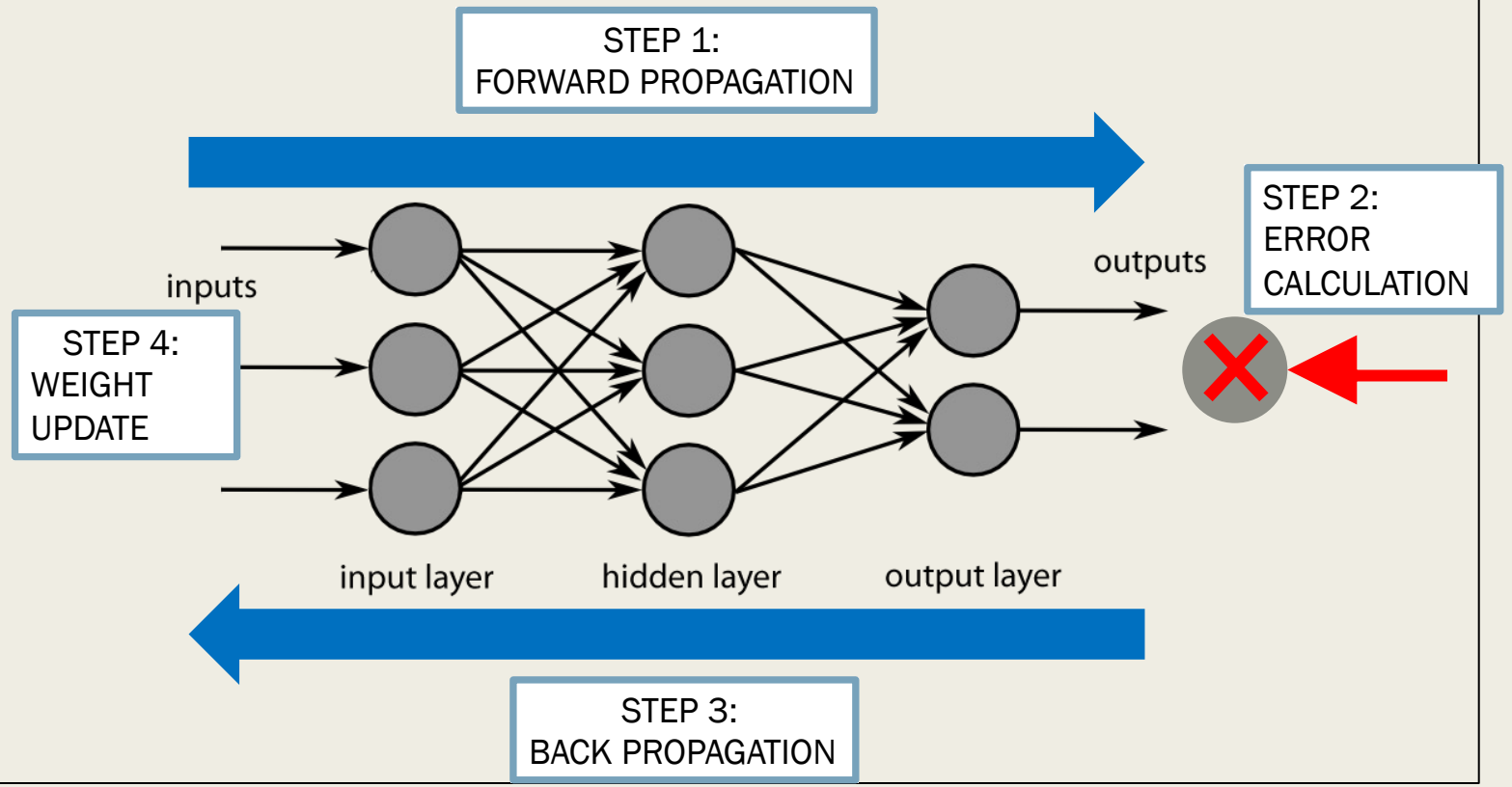



01 - Regression

# TRAINING A NETWORK

# Back Propagation

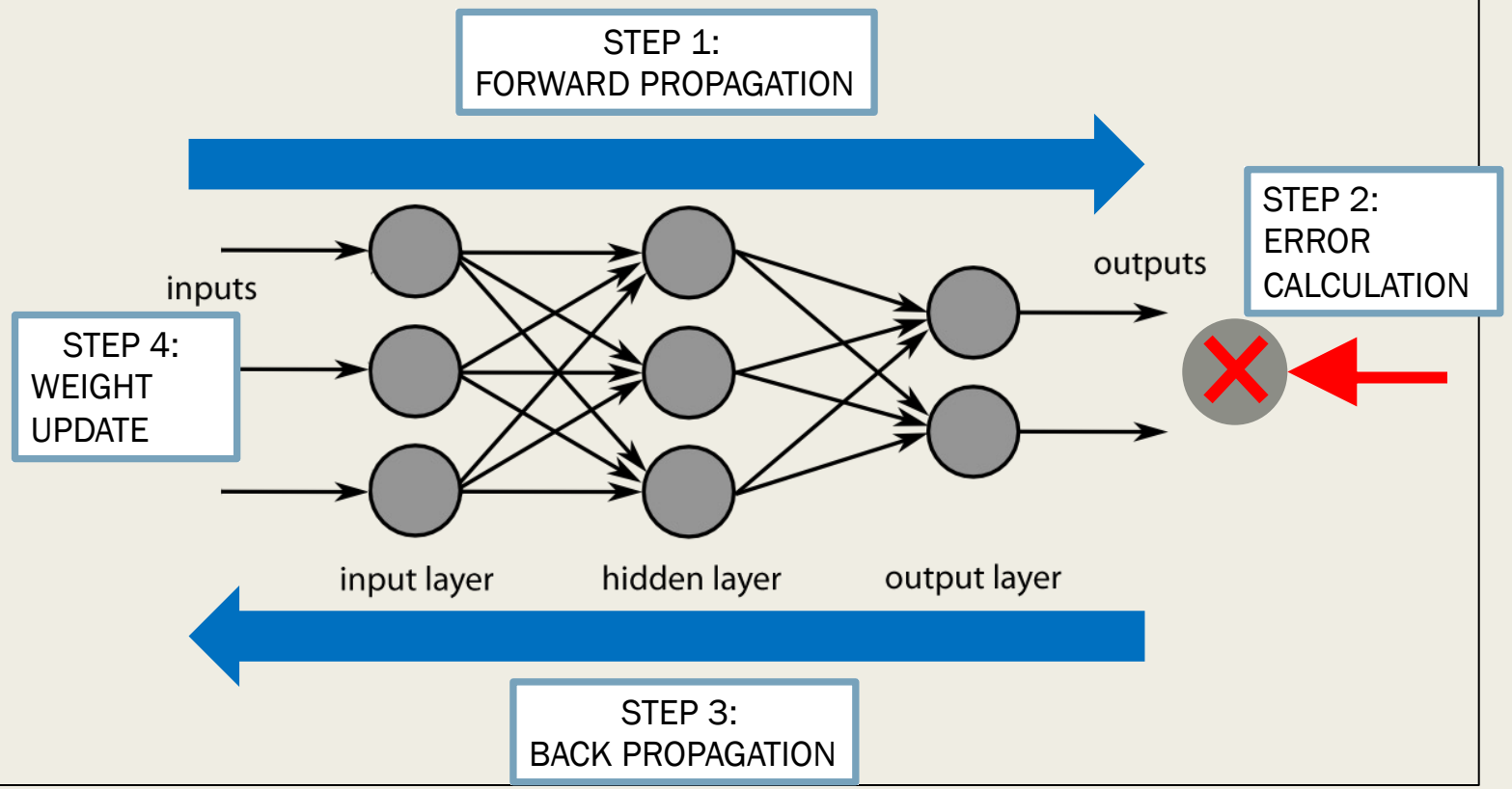
A method used to train ANNs by calculating gradient needed to update network weights.




Often used by the gradient descent optimization algorithm to adjust the weight of neurons by calculating the gradient of the loss function.

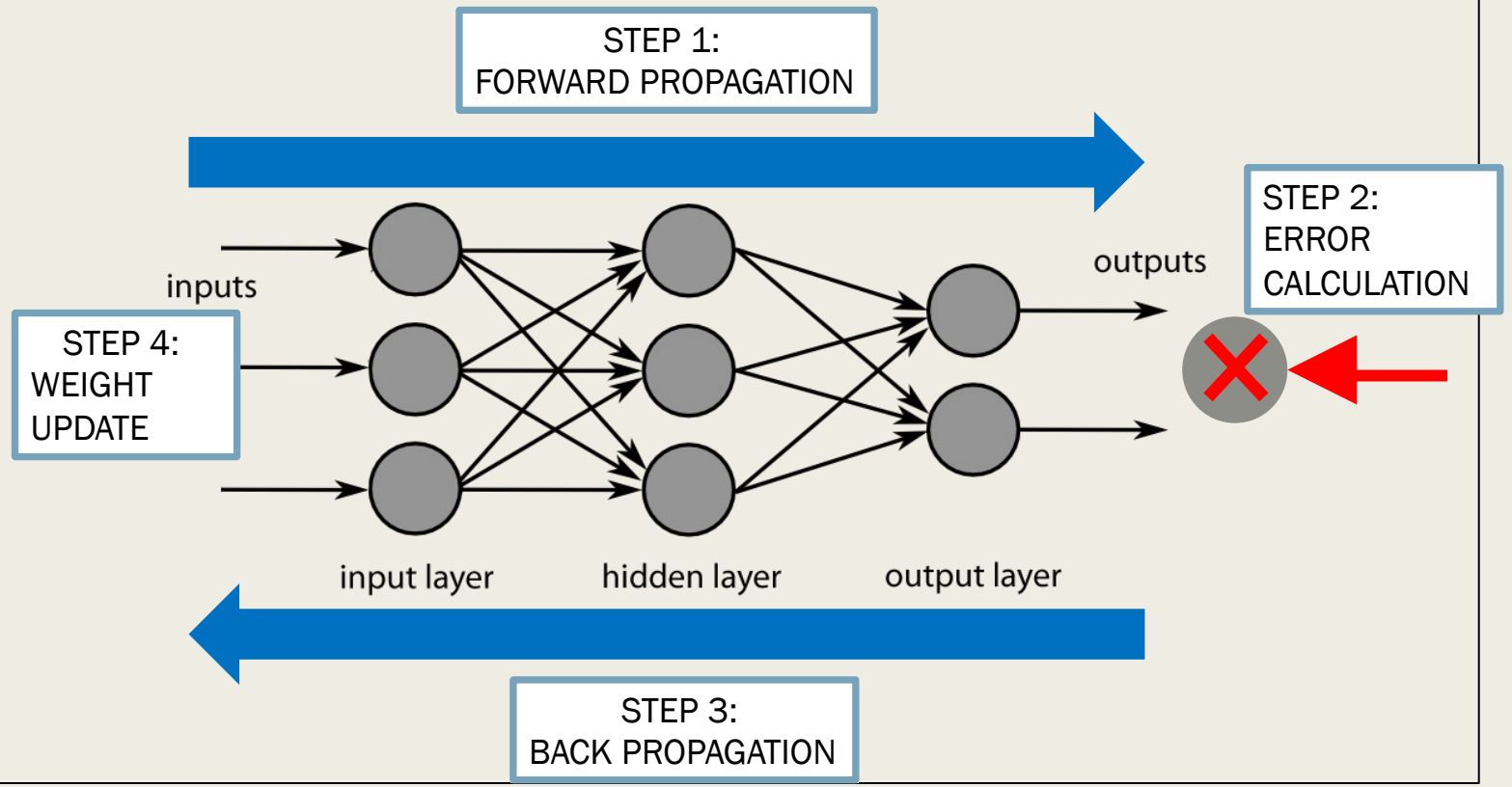
# Back Propagation – Phase 1 - Propagation

1. Propagation forward through the network to generate the output value(s)
2. Calculation of the cost (error term)
3. Propagation of output activations back through network using training pattern target in order to generate the deltas (difference between targeted and actual output values)



# Back Propagation – Phase 2 – Weight Update

1. Calculate weight gradient
2. A ratio (percentage) of the weight's gradient is subtracted from the weight.
3. This ratio influences the speed and quality of learning and called learning rate. The greater the ratio, the faster neuron train, but lower ratio, more accurate the training is.



# MULTI NEURON MODEL

# 2 Neurons

The network is represented by a matrix of weights, inputs and outputs.

Total Number of adjustable parameters = 8:

*Weights = 6*

*Biases = 2*

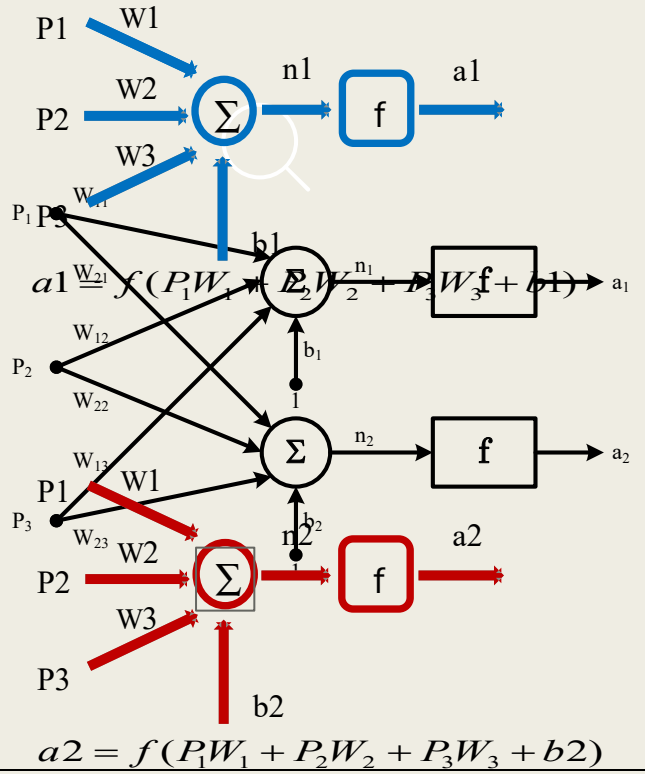
Matrix Representation

$$P = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

$$W = \begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \end{bmatrix}$$

$$b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

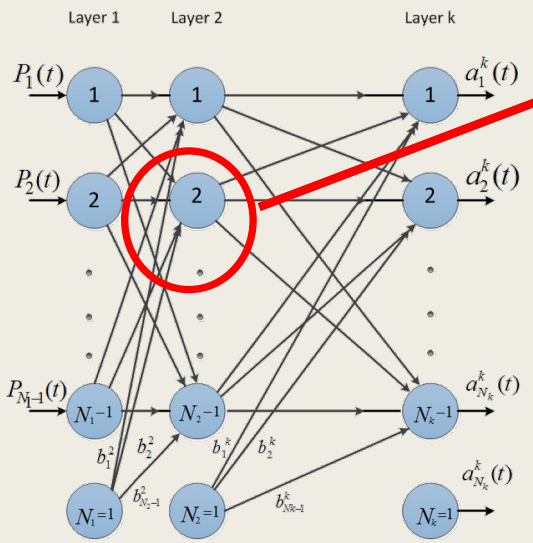
$$a = f(W \times P + b)$$

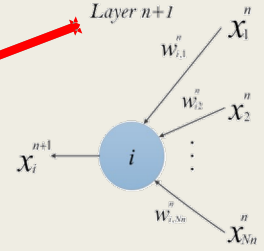


# Multi Neuron Network - Matrices

$$P = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_{N_1} \end{bmatrix}$$

$$\begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1,N_1} \\ W_{21} & W_{22} & & W_{2,N_1} \\ \vdots & & \ddots & \vdots \\ W_{m-1,1} & W_{m-1,2} & \cdots & W_{m-1,N_1} \\ W_{m,1} & W_{m,2} & & W_{m,N_1} \end{bmatrix}$$




$$x_i^{n+1}(t) = \varphi\left(\sum_{j=1}^{N_n} w_{i,j}^n \, x_j^n(t)\right)$$

Node *(n+1, i)* representation

**Non-Linear Sigmoid Activation function**

$$\varphi(w) = \frac{1}{1 + e^{-w}}$$

*m: number of neurons in the hidden layer*

*$N_1$: number of inputs*

# Questions