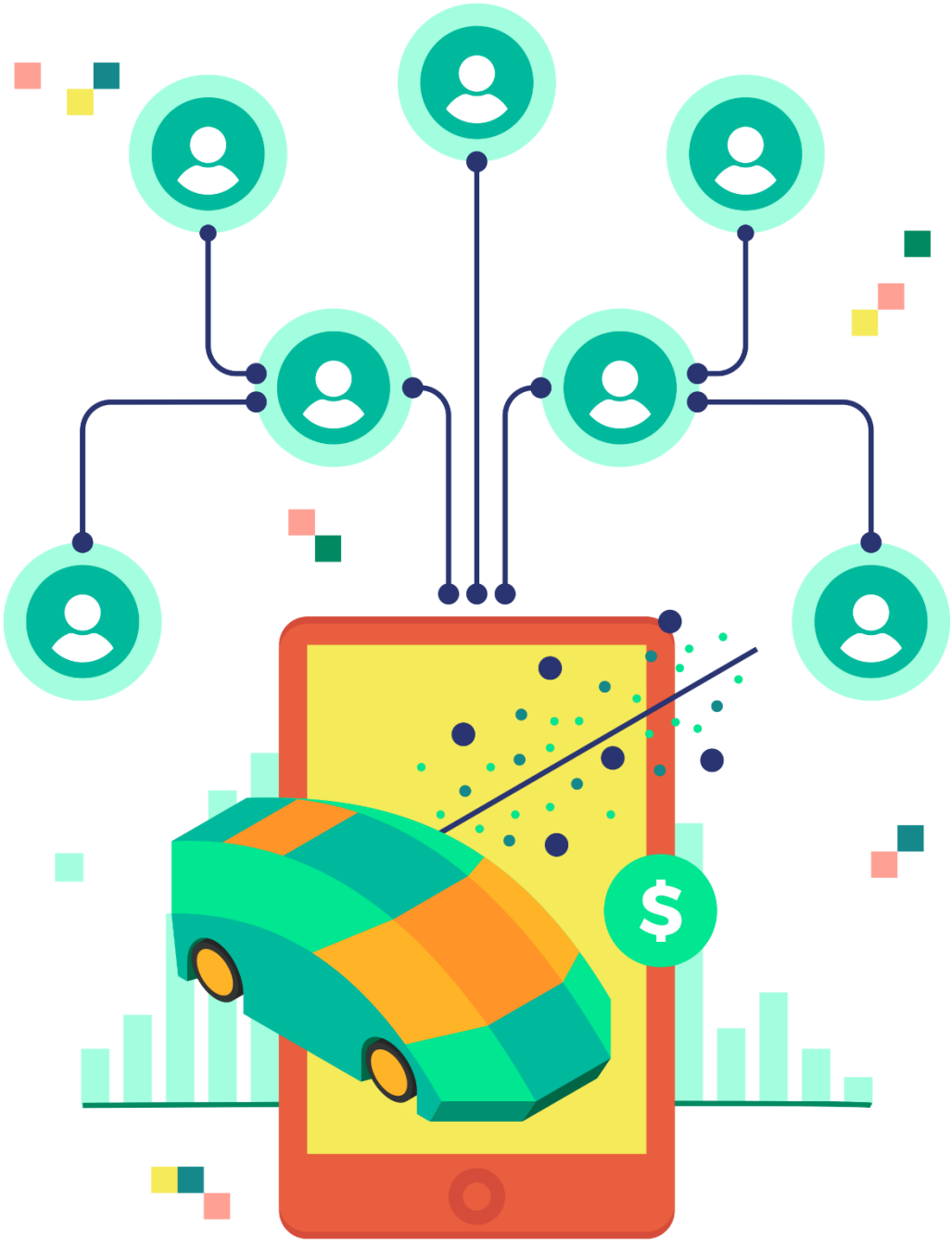
DEEP AND MACHINE LEARNING **PRACTICAL**

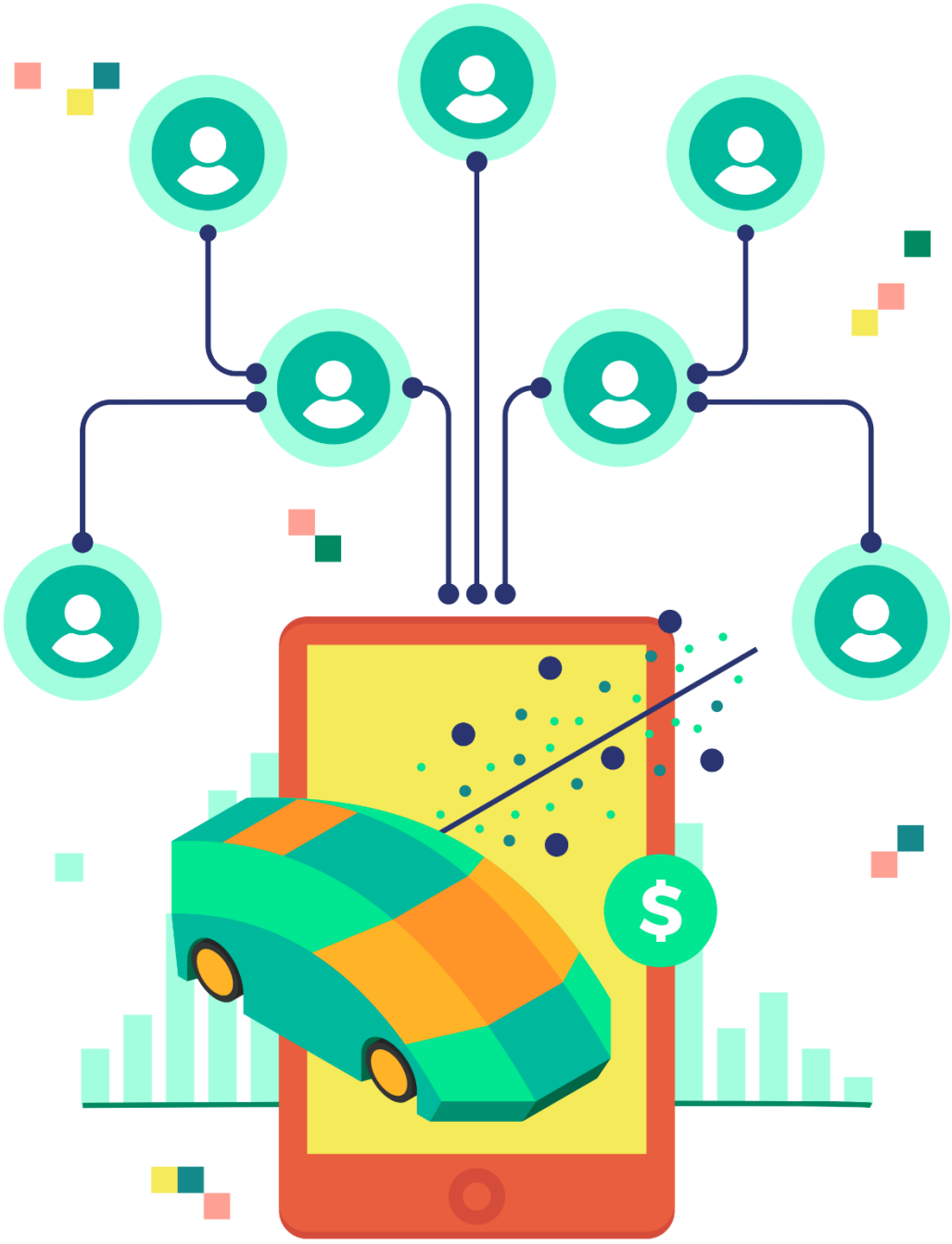
SuperDataScience

ARTIFICIAL NEURAL NETWORKS

CAR SALES PREDICTION



PROBLEM STATEMENT & REGRESSION BASICS



PROJECT OVERVIEW

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

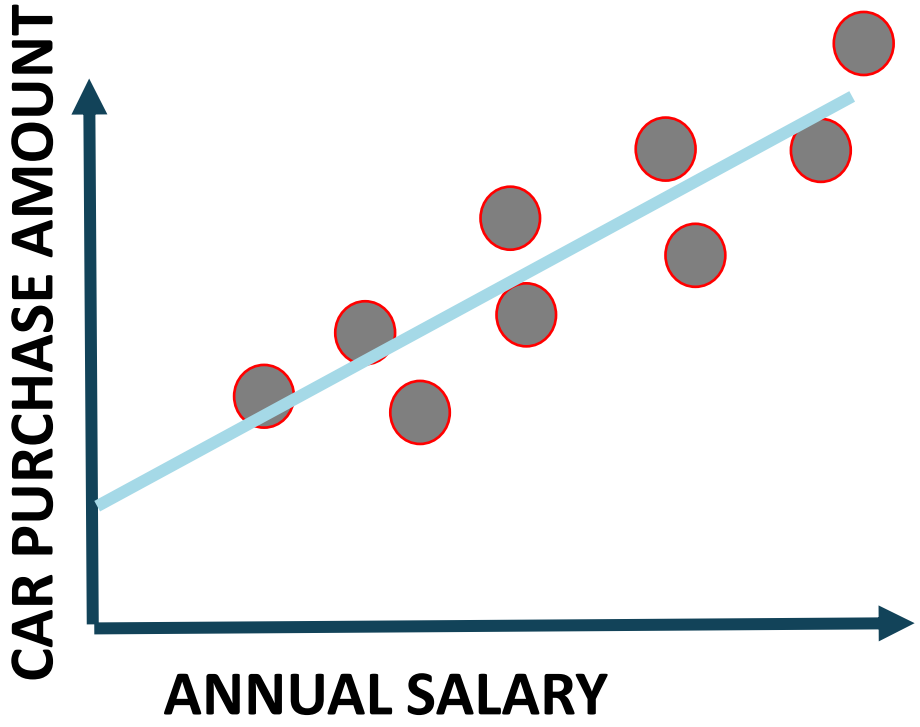
REGRESSION TASK!

The model should predict:

- Car Purchase Amount

WHAT IS REGRESSION? INTUITION

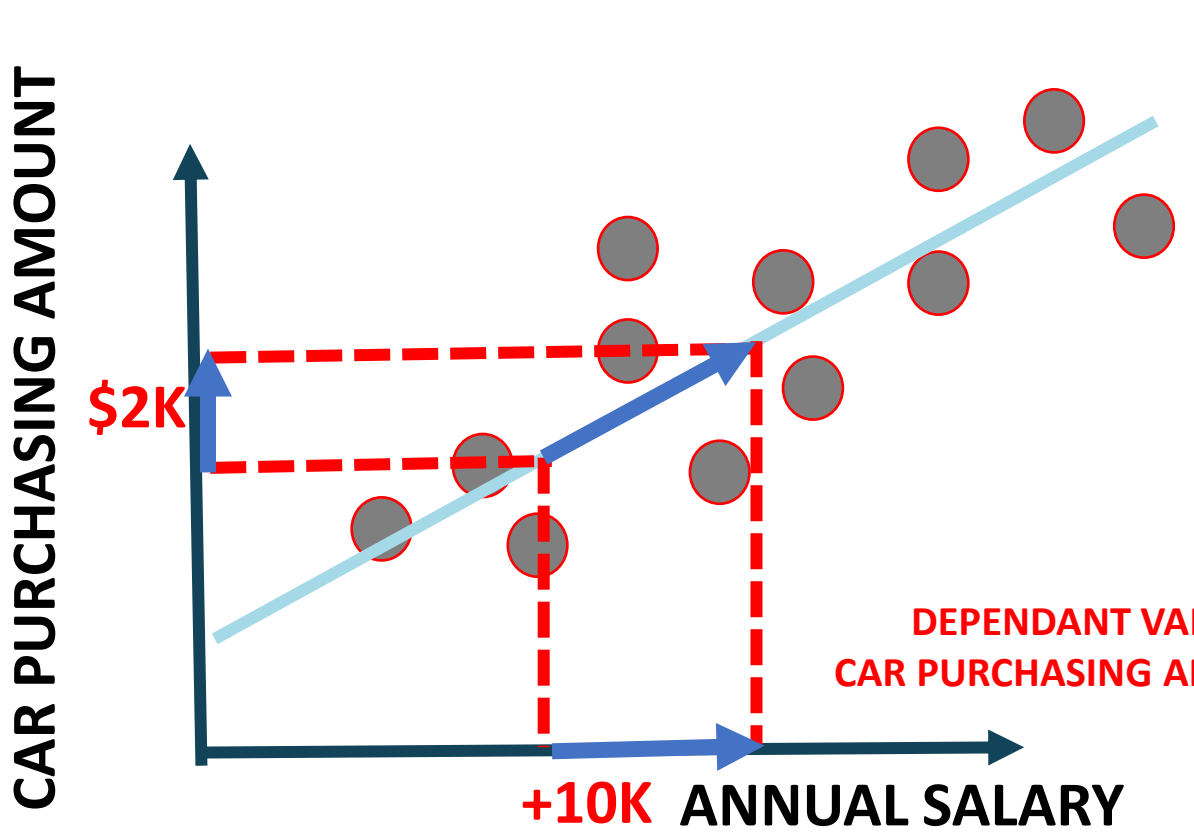
- Regression works by predicting value of one variable Y based on another variable X.
- X is called the independent variable and Y is called the dependant variable.



Annual Salary	Car Purchase Amount
62812.09301	35321.45877
66646.89292	45115.52566
53798.55112	42925.70921
79370.03798	67422.36313
59729.1513	55915.46248
68499.85162	56611.99784
39814.522	28925.70549
51752.23445	47434.98265
58139.2591	48013.6141
53457.10132	38189.50601
73348.70745	59045.51309
55421.65733	42288.81046
37336.3383	28700.0334
68304.47298	49258.87571
72776.00382	49510.03356
64662.30061	53017.26723
63259.87837	41814.72067
52682.06401	43901.71244
54503.14423	44633.99241
55368.23716	54827.52403
63435.86304	51130.95379
64347.34531	43402.31525

REGRESSION MATH!

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



MODEL! (GOAL)

$$y = b + aX$$

DEPENDANT VARIABLE
CAR PURCHASING AMOUNT ($)

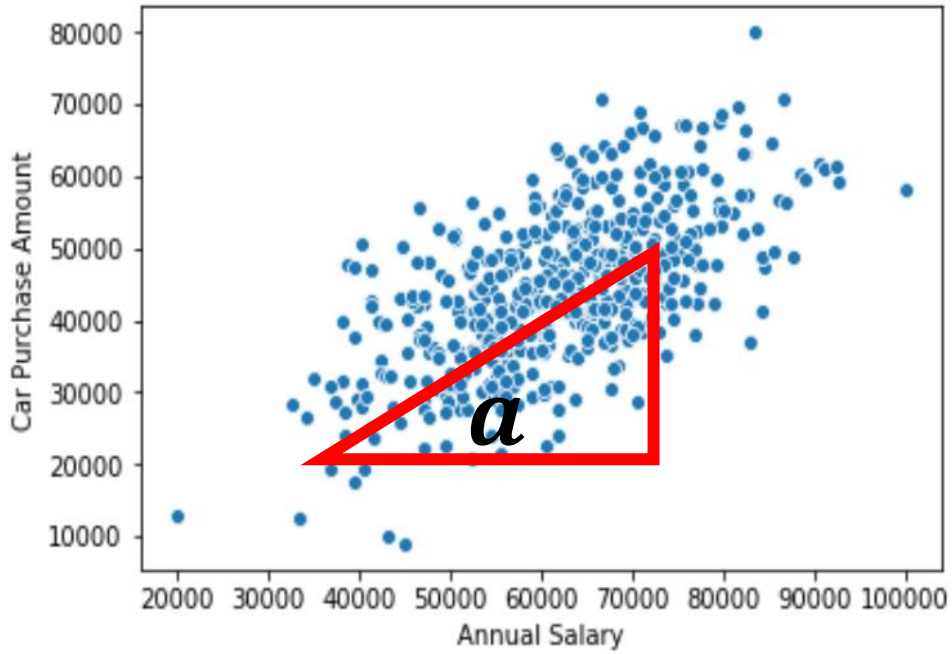
INDEPENDENT VARIABLE
ANNUAL SALARY ($)

HOW ARE WE GOING TO USE THE MODEL?

- Once the coefficients '**m**' and '**b**' are obtained, you have obtained a regression model!
- This "trained" model can be later used to predict car purchase amount (dollars) based on the annual salary

Y

b



X

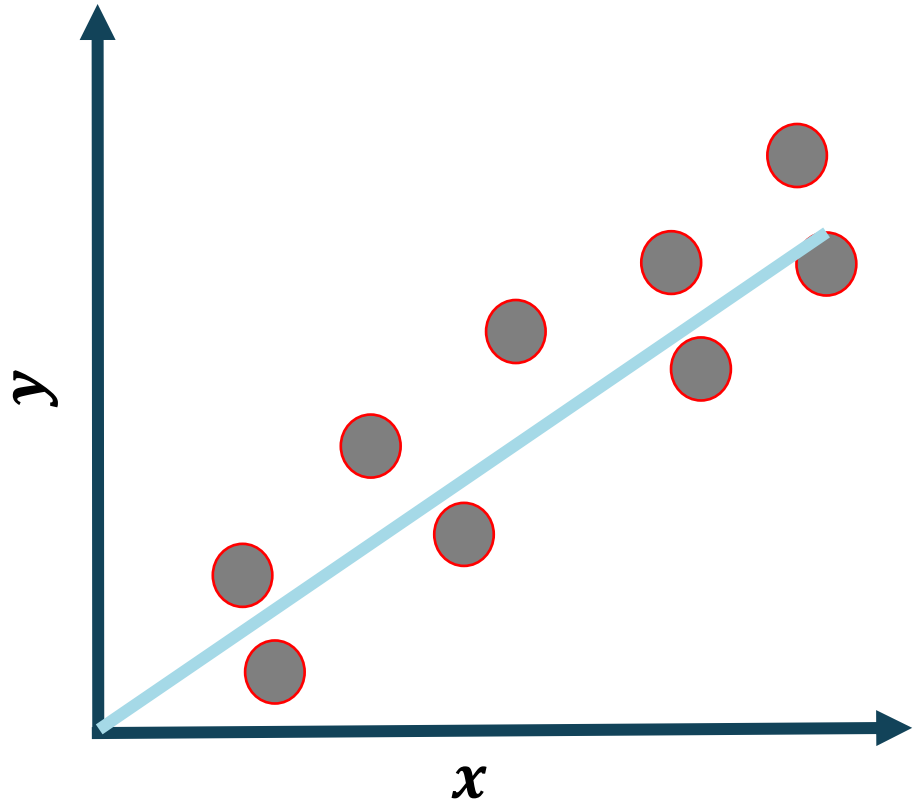
$y = mX + b$

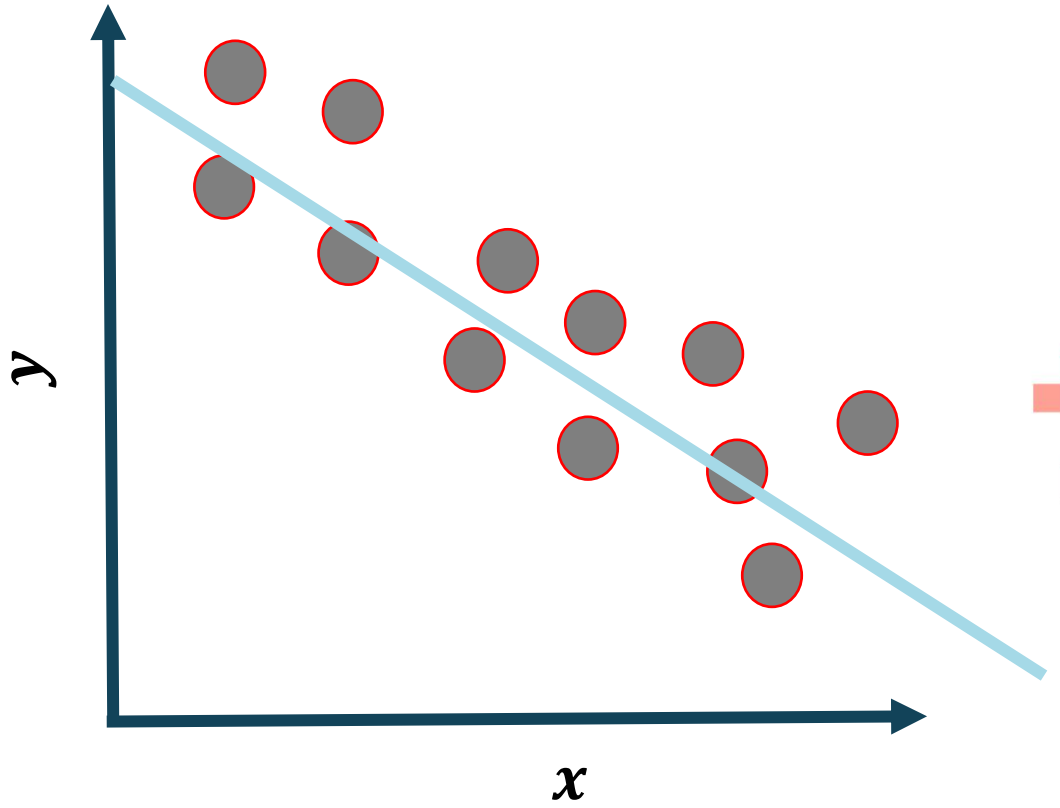
DEPENDANT VARIABLE **INDEPENDANT VARIABLE**

REGRESSION QUIZ

- Match the equations to the figures:

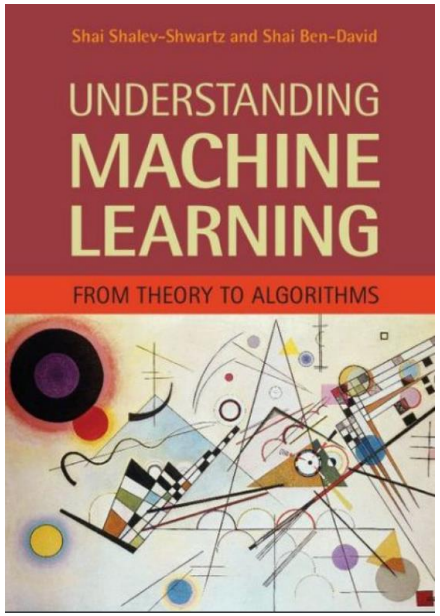
$y = 3 * x$ $y = 15 - 10 * x$



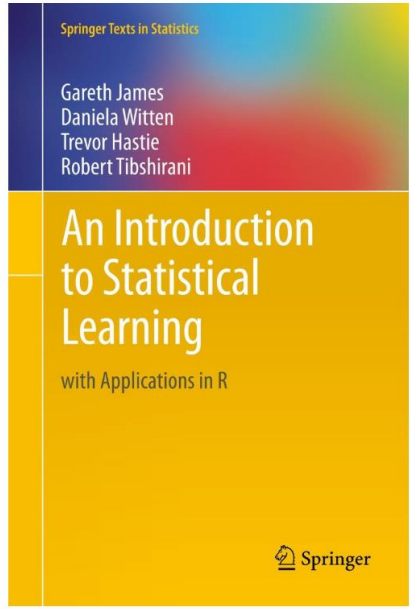


ADDITIONAL READING MATERIAL

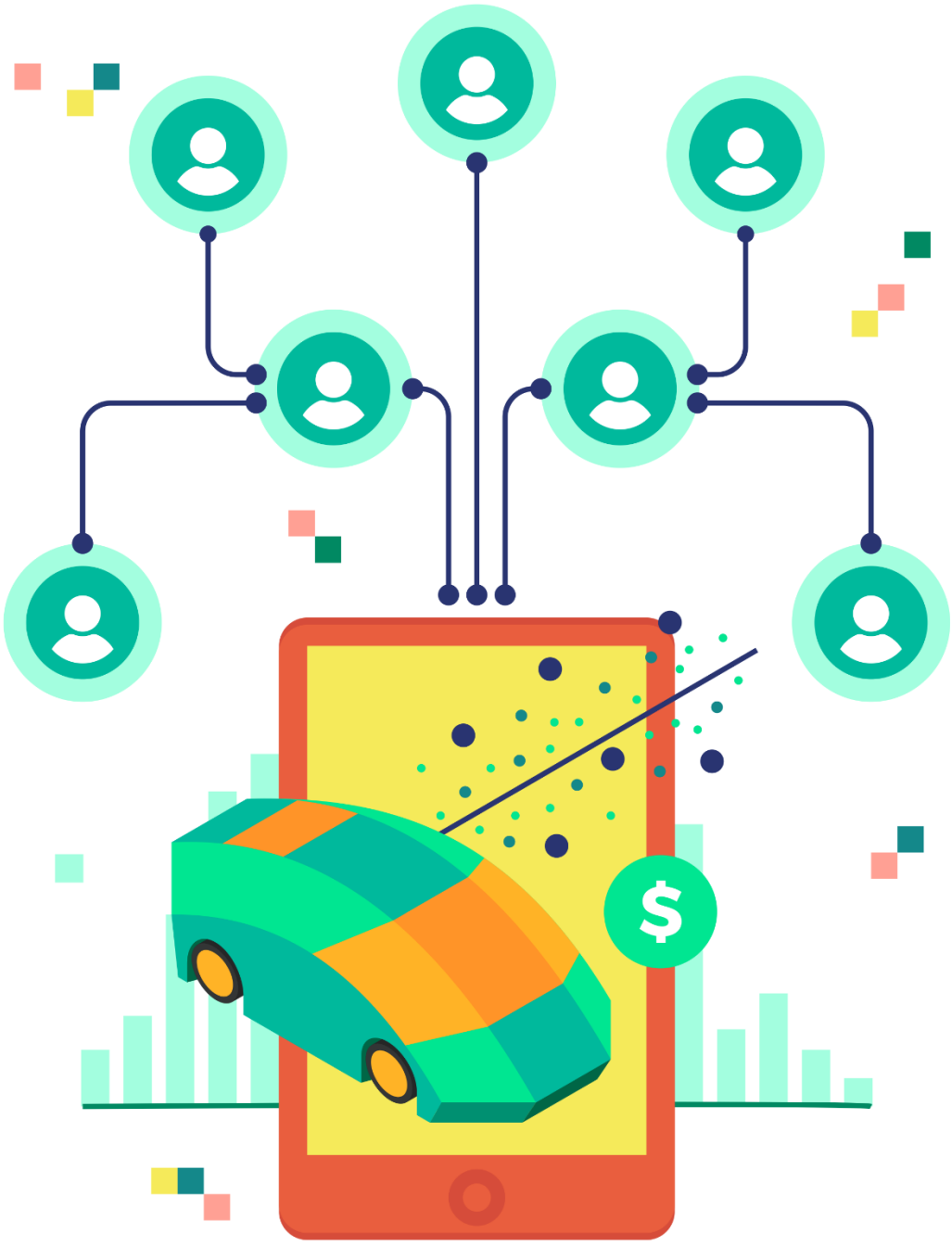
Additional Resources, Page #123:
http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf



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

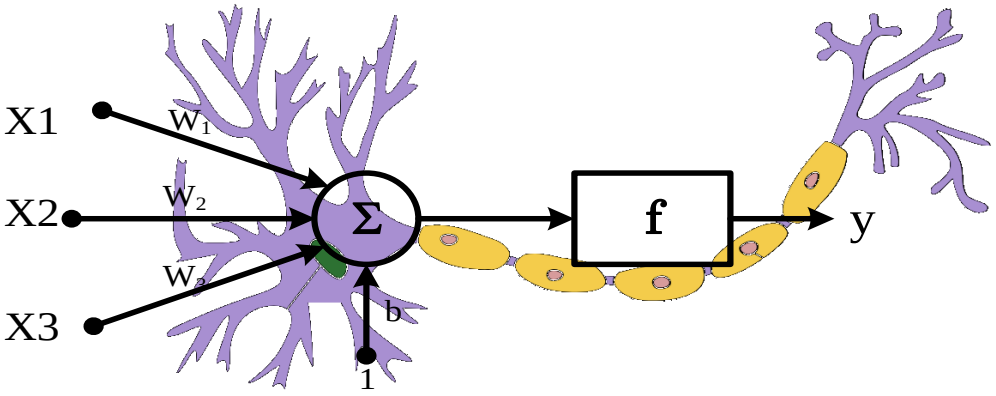


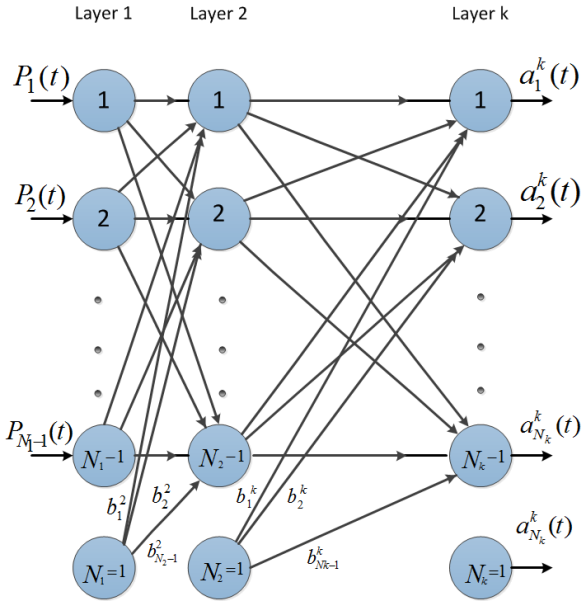
WHAT ARE ANNs AND HOW DO THEY LEARN?



ARTIFICIAL NEURAL NETWORK: 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.





- **Photo Credit:** https://commons.wikimedia.org/wiki/File:Neuron_Hand-tuned.svg

HOW DO HUMANS LEARN? FROM EXPERIENCE

- Humans learn from experience (by example)

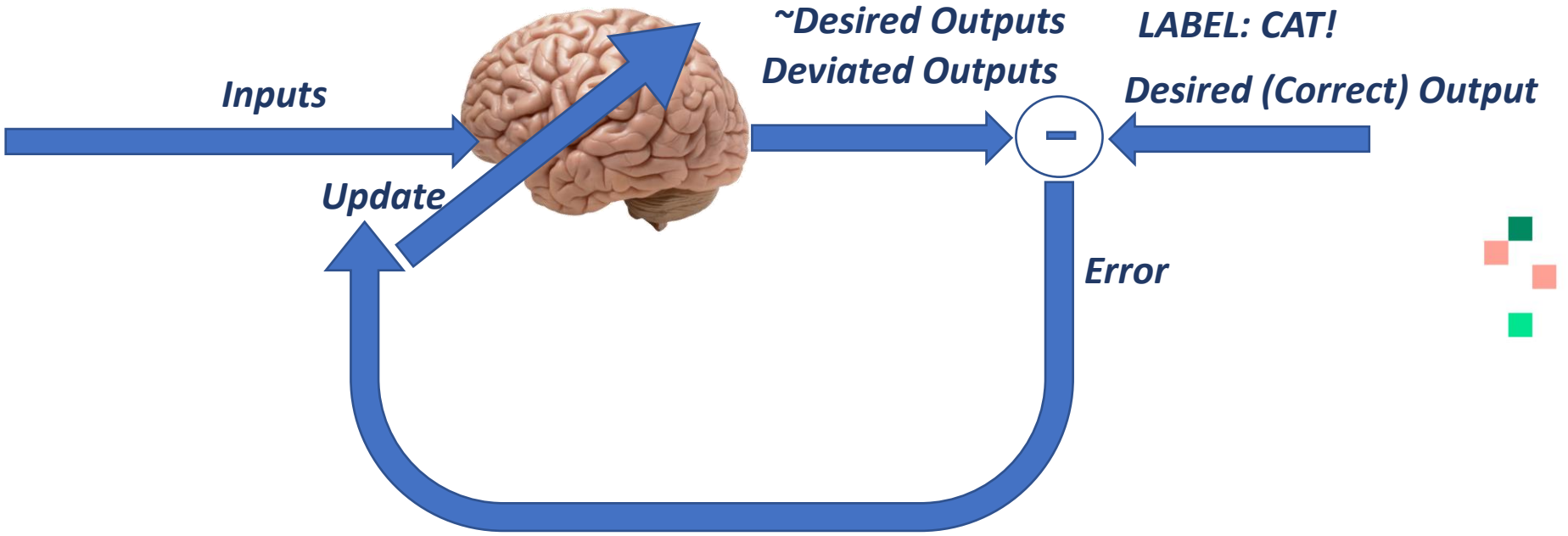
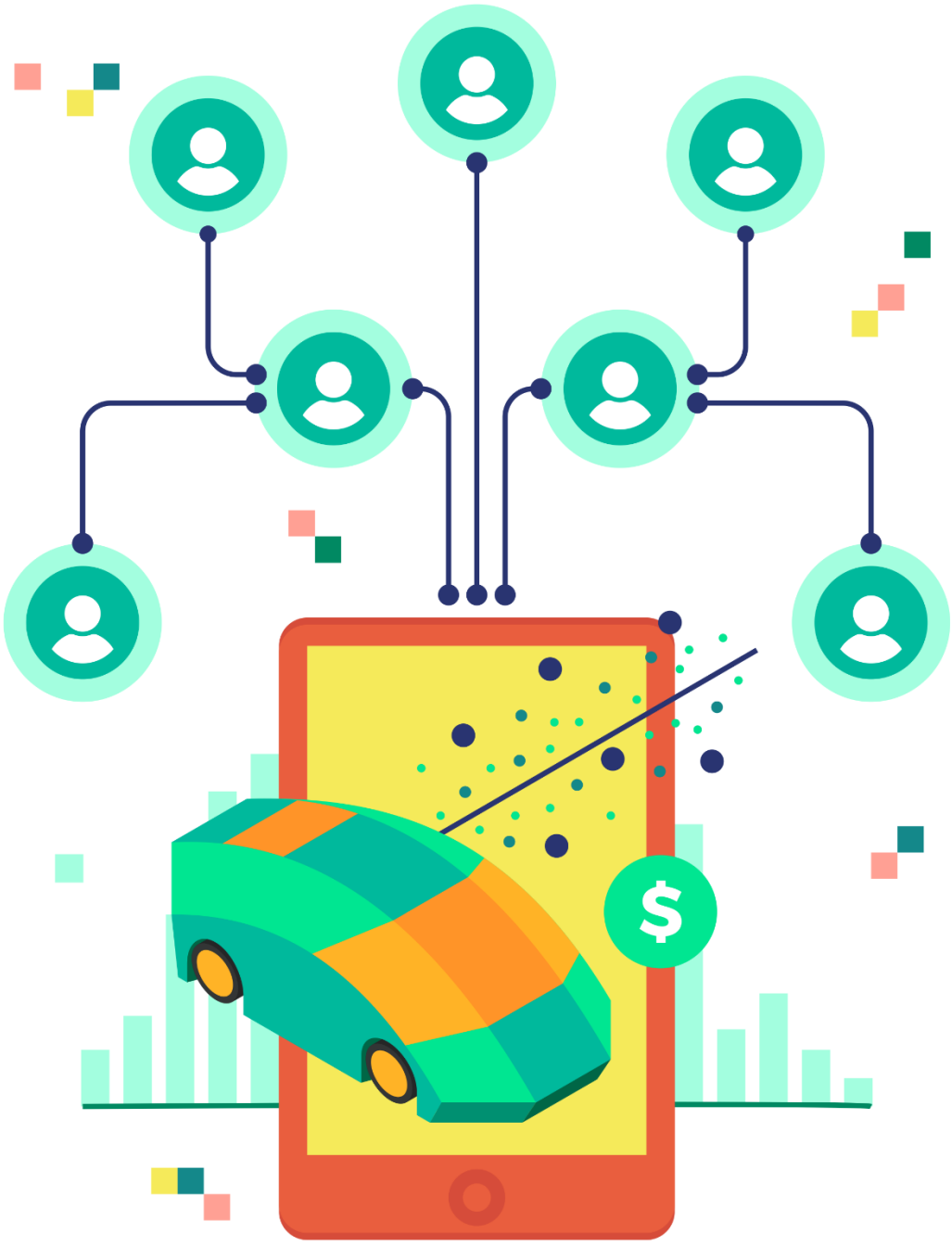


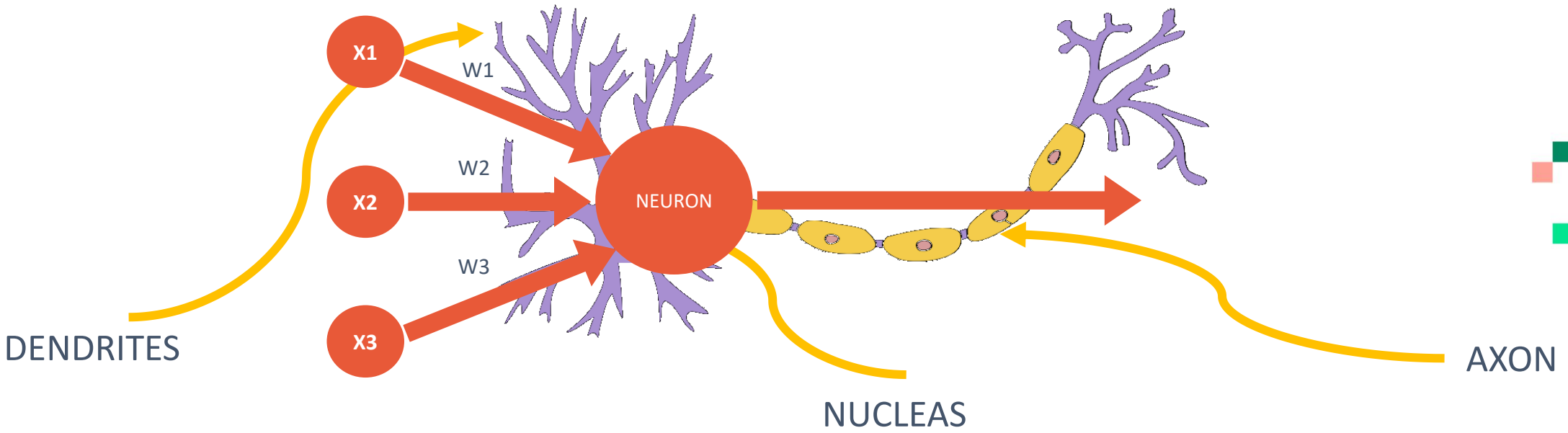
Photo Credit: https://www.flickr.com/photos/flamephoenix1991/8376271918
Photo Credit: https://www.pexels.com/photo/grey-and-white-short-fur-cat-104827/

SINGLE NEURON MODEL



NEURON MATHEMATICAL MODEL

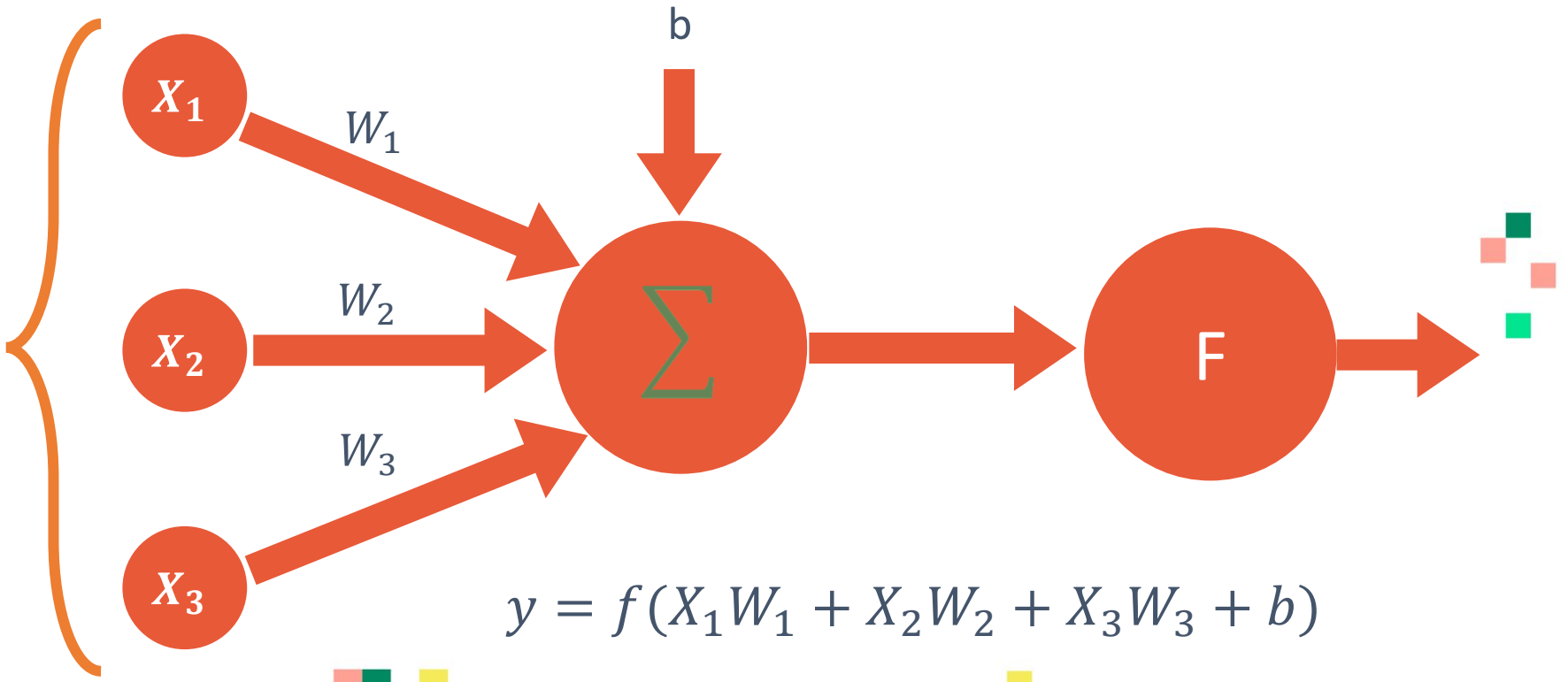
- The neuron collects signals from input channels named dendrites, processes information in its nucleus, and then generates an output in a long thin branch called axon.



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”.

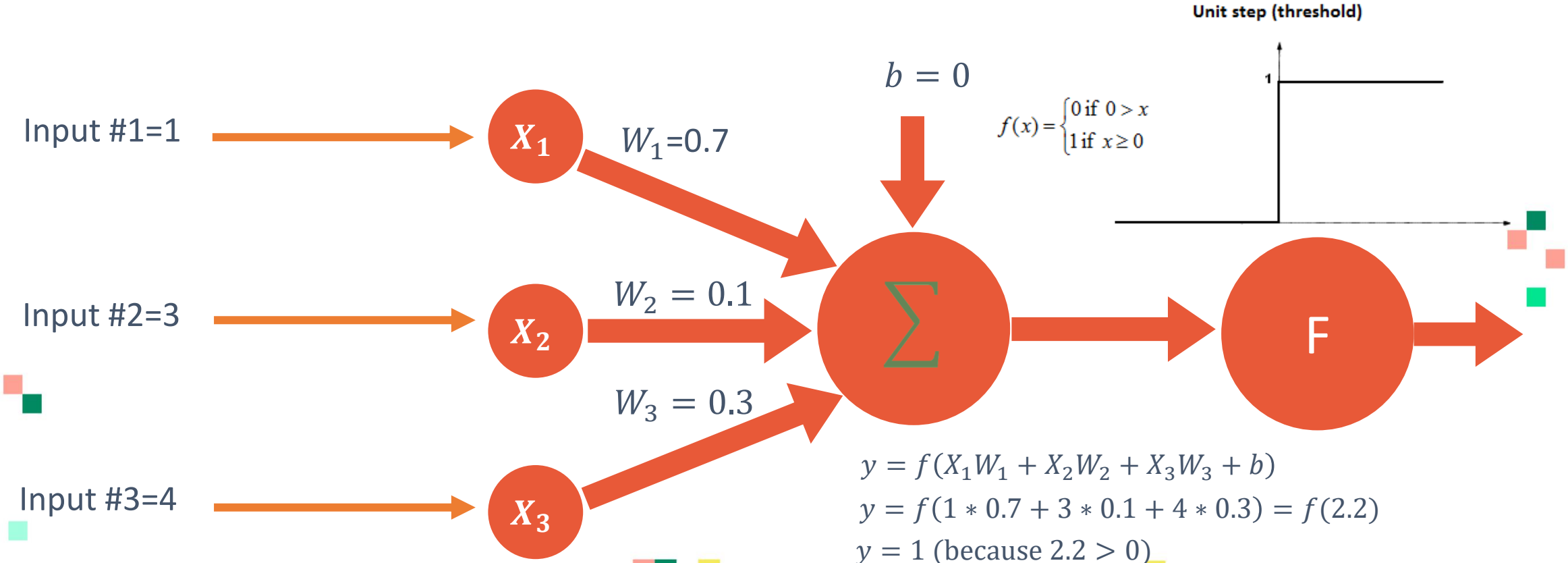
INPUTS/INDEPENDENT VARIABLES



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

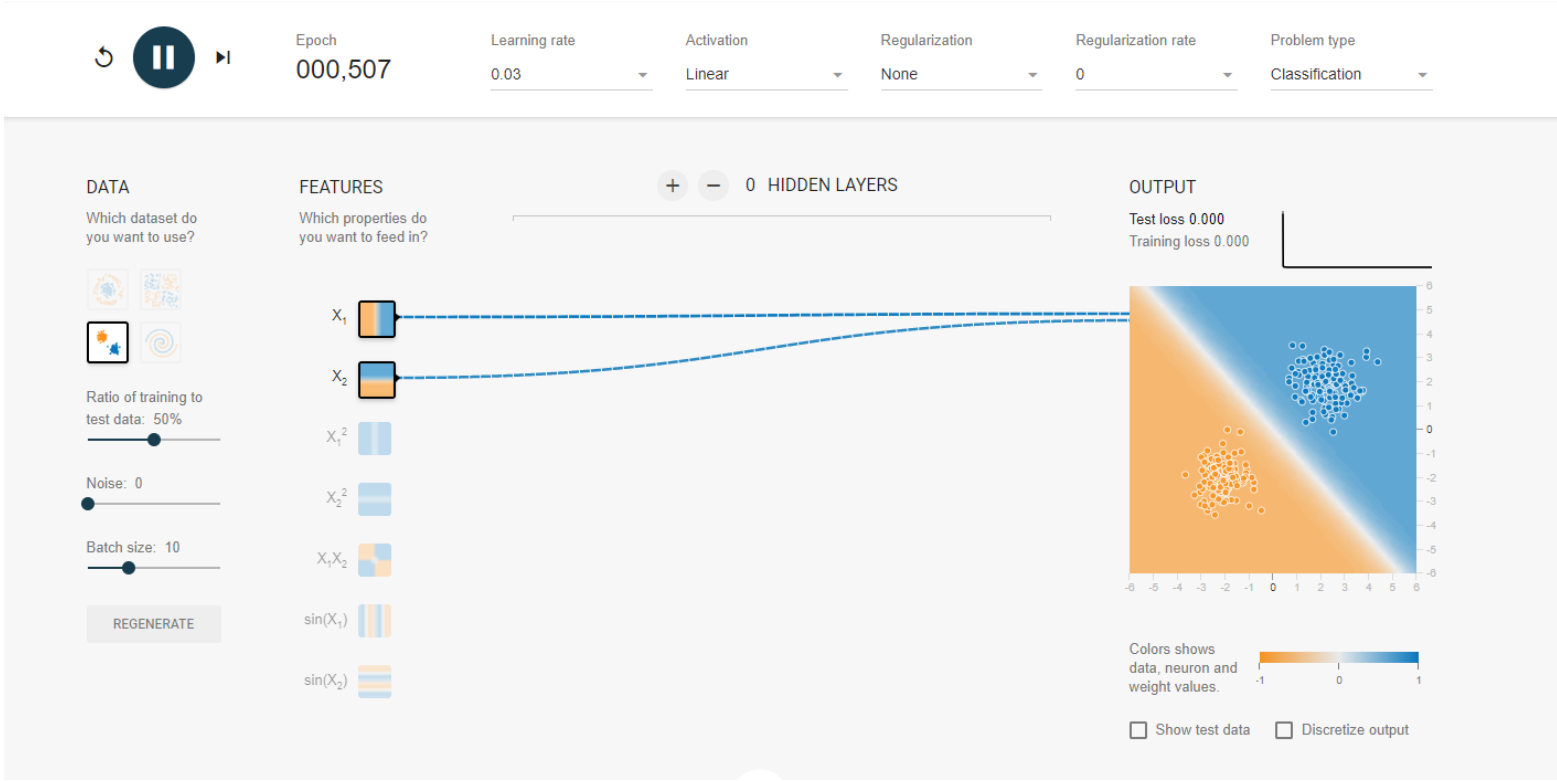
SINGLE NEURON MODEL IN ACTION

- Let's assume that the activation function is a Unit Step Activation Function.
- The activation functions is used to map the input between (0, 1).

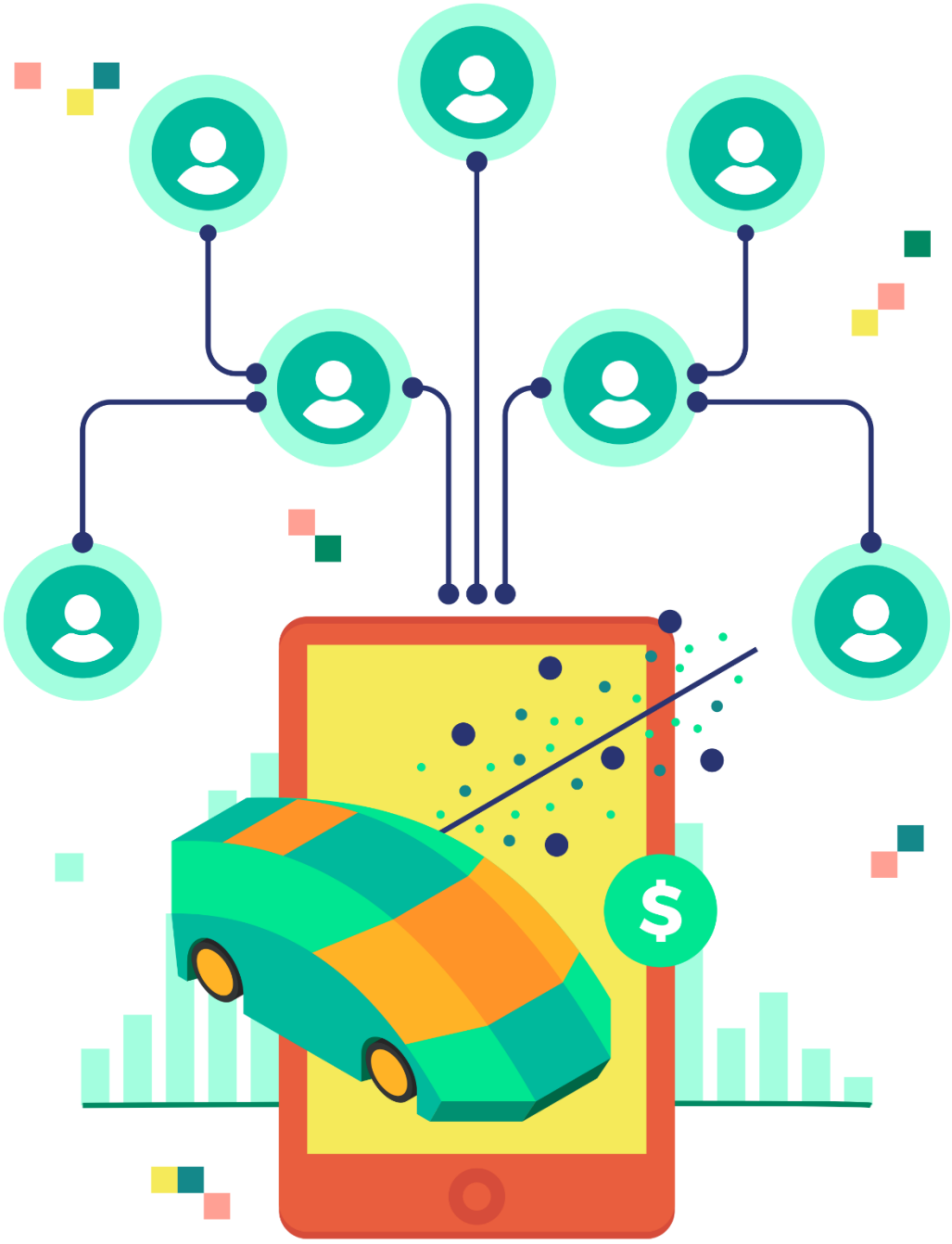


SINGLE NEURON MODEL IN ACTION

- Check this out: https://playground.tensorflow.org

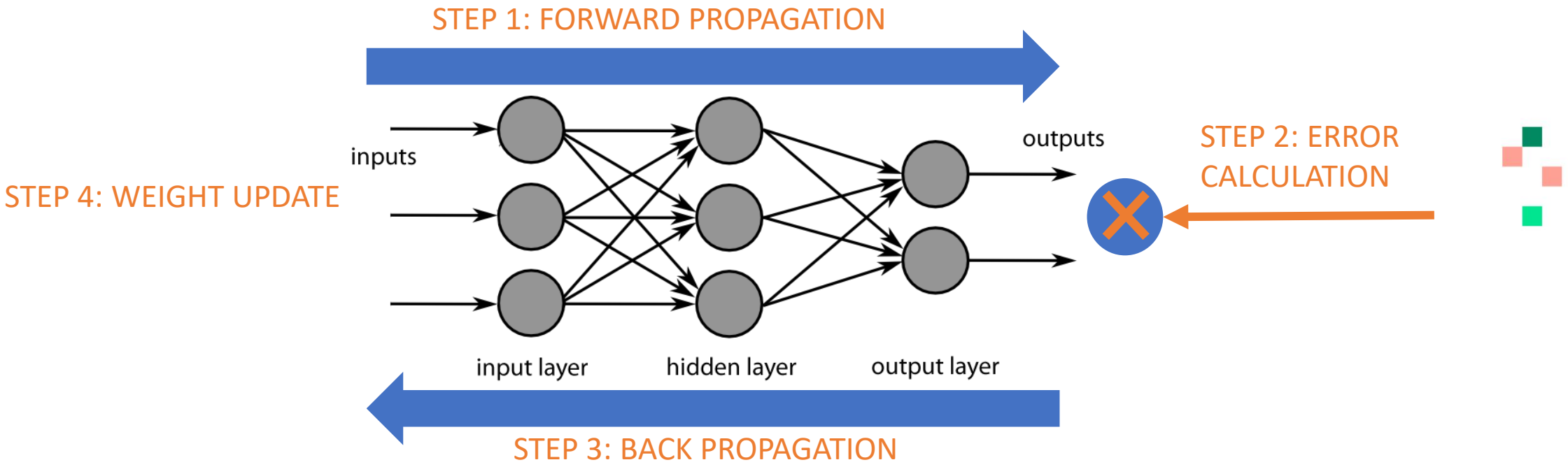


NETWORK TRAINING



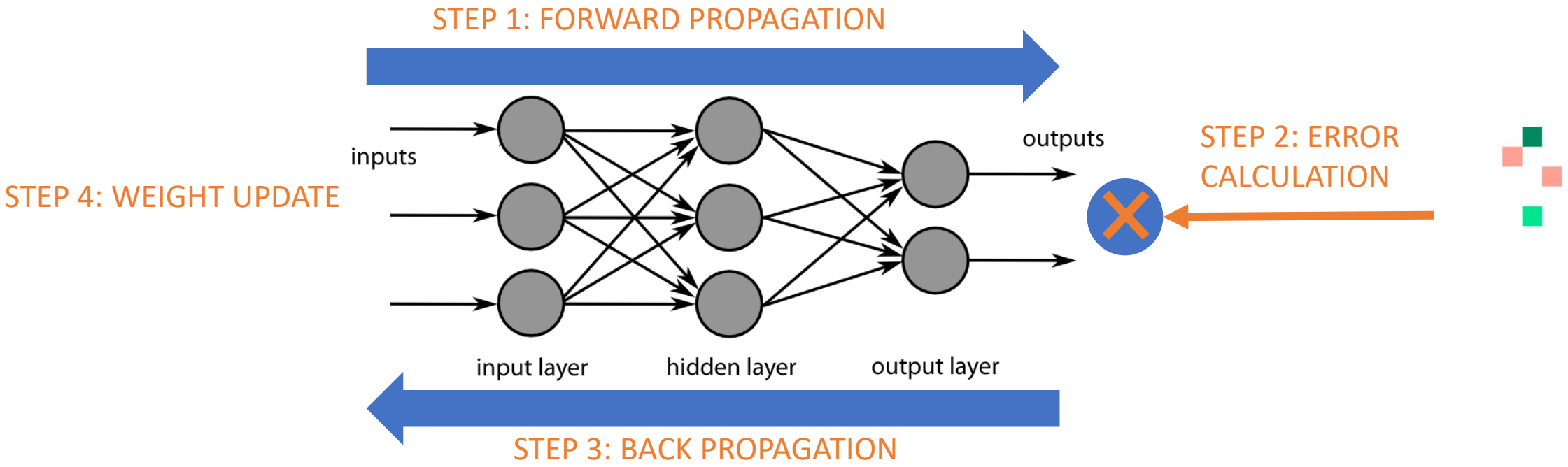
NETWORK TRAINING: BACK PROPAGATION

- Backpropagation is a method used to train ANNs by calculating gradient needed to update network weights.
- It is commonly used by the gradient descent optimization algorithm to adjust the weight of neurons by calculating the gradient of the loss function.



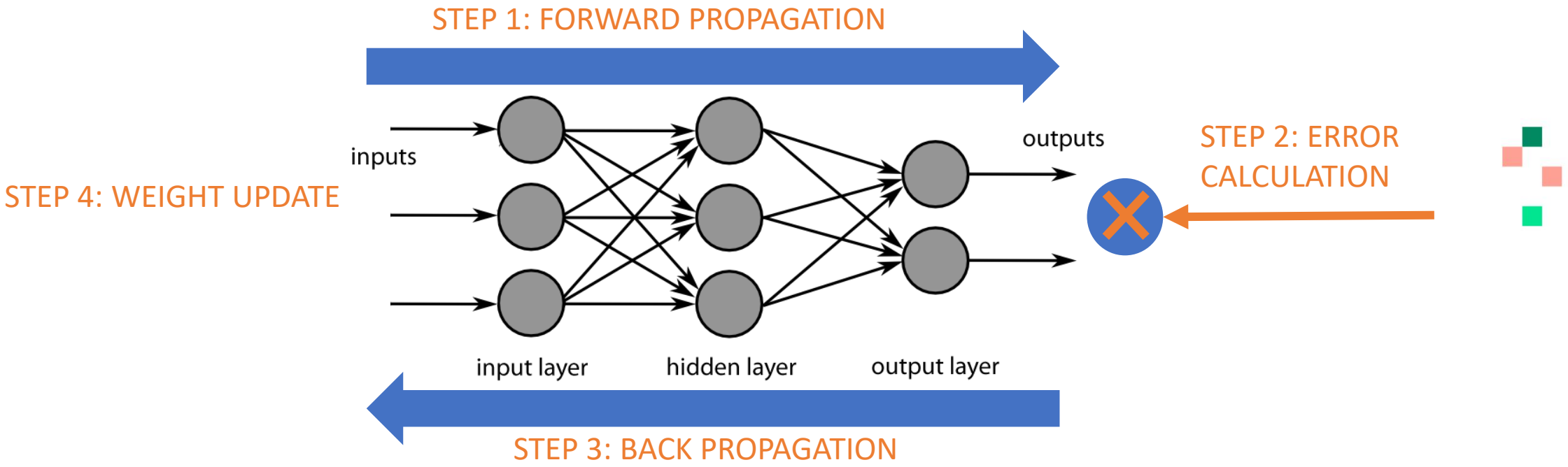
NETWORK TRAINING: BACK PROPAGATION

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



NETWORK TRAINING: BACK PROPAGATION

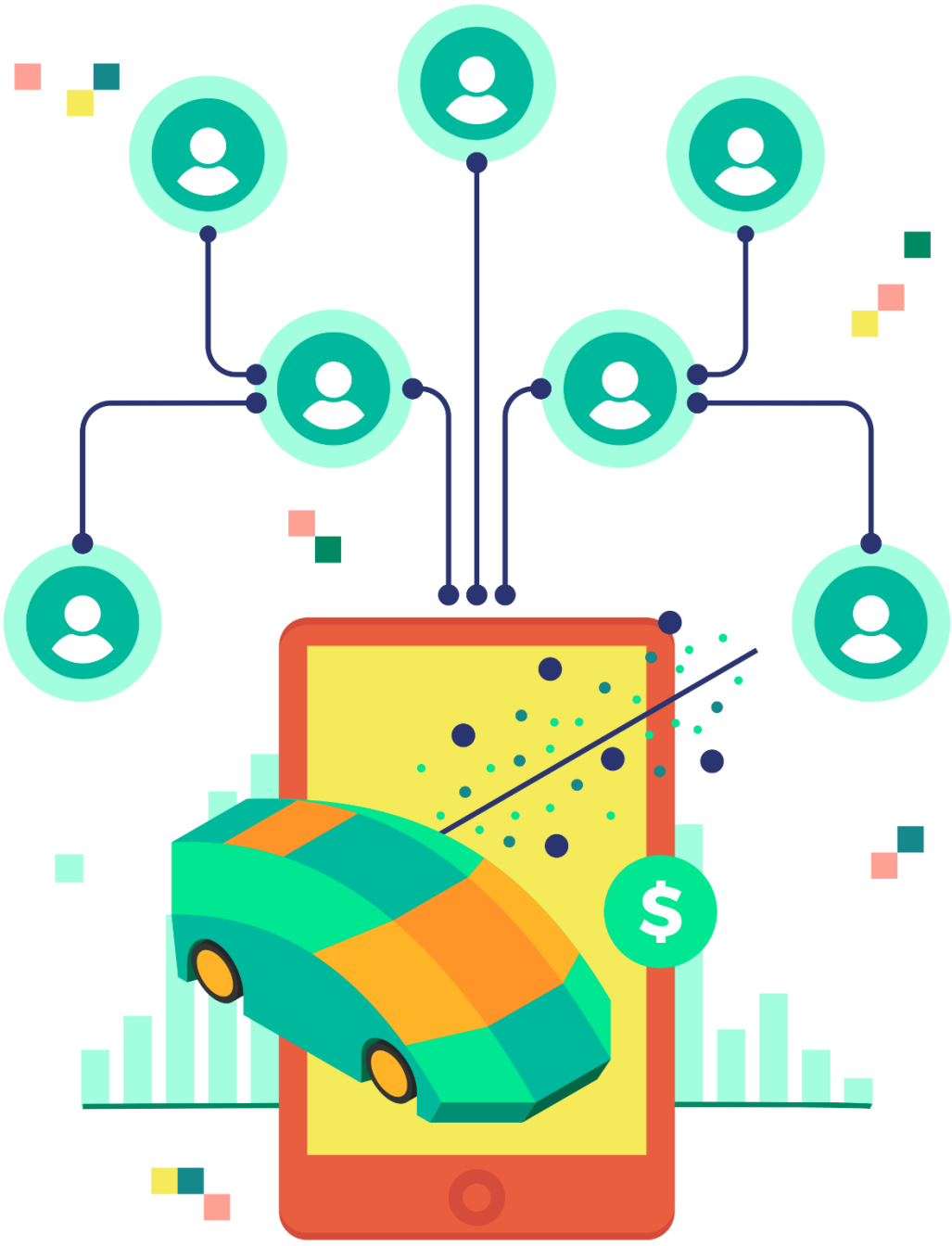
- Phase 2: weight update
  - Calculate weight gradient.
  - A ratio (percentage) of the weight's gradient is subtracted from the weight.
  - This ratio influences the speed and quality of learning and called learning rate. The greater the ratio, the faster neuron train, but lower ratio, more accurate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



READING MATERIALS: BACK PROPAGATION

- “Backpropagation neural networks: A tutorial” by Barry J.Wythoff
- “Improved backpropagation learning in neural networks with windowed momentum”, International Journal of Neural Systems, vol. 12, no.3&4, pp. 303-318.

MULTI- NEURON MODEL



TWO NEURON MODEL: MATRIX REPRESENTATION

- The network is represented by a matrix of weights, inputs and outputs.
- Total Number of adjustable parameters = 8:
  - Weights = 6
  - Biases = 2

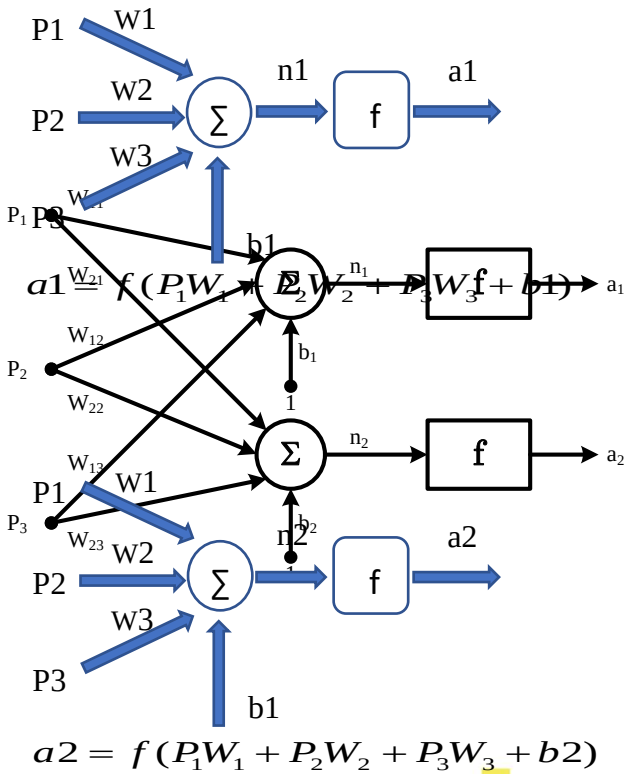
Matrix Representation

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

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

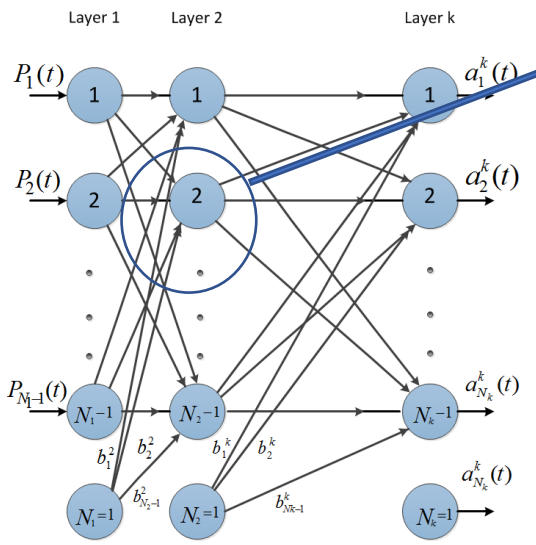
$$b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

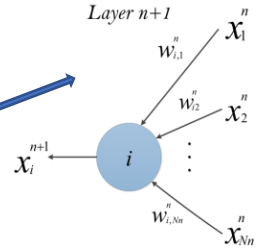
$$a = f(W \times P + b)$$



MULTI-LAYER PERCEPTRON MODEL: MATRIX REPRESENTATION

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





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

Node *(n+1, i)* representation

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

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