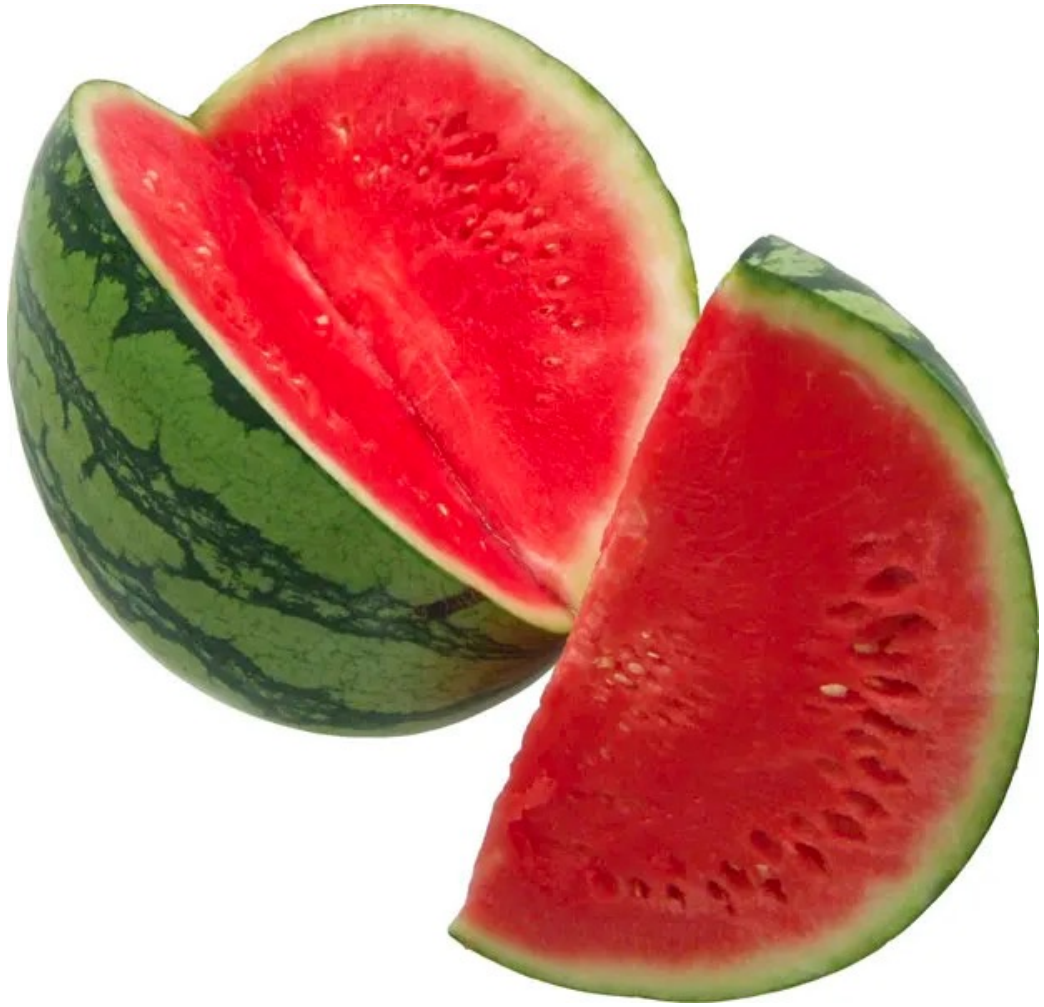
Neural Network Workshop

Khayyam Salehi, Ph.D.
Assistant Professor of Computer Science
Shahrekord University

2023

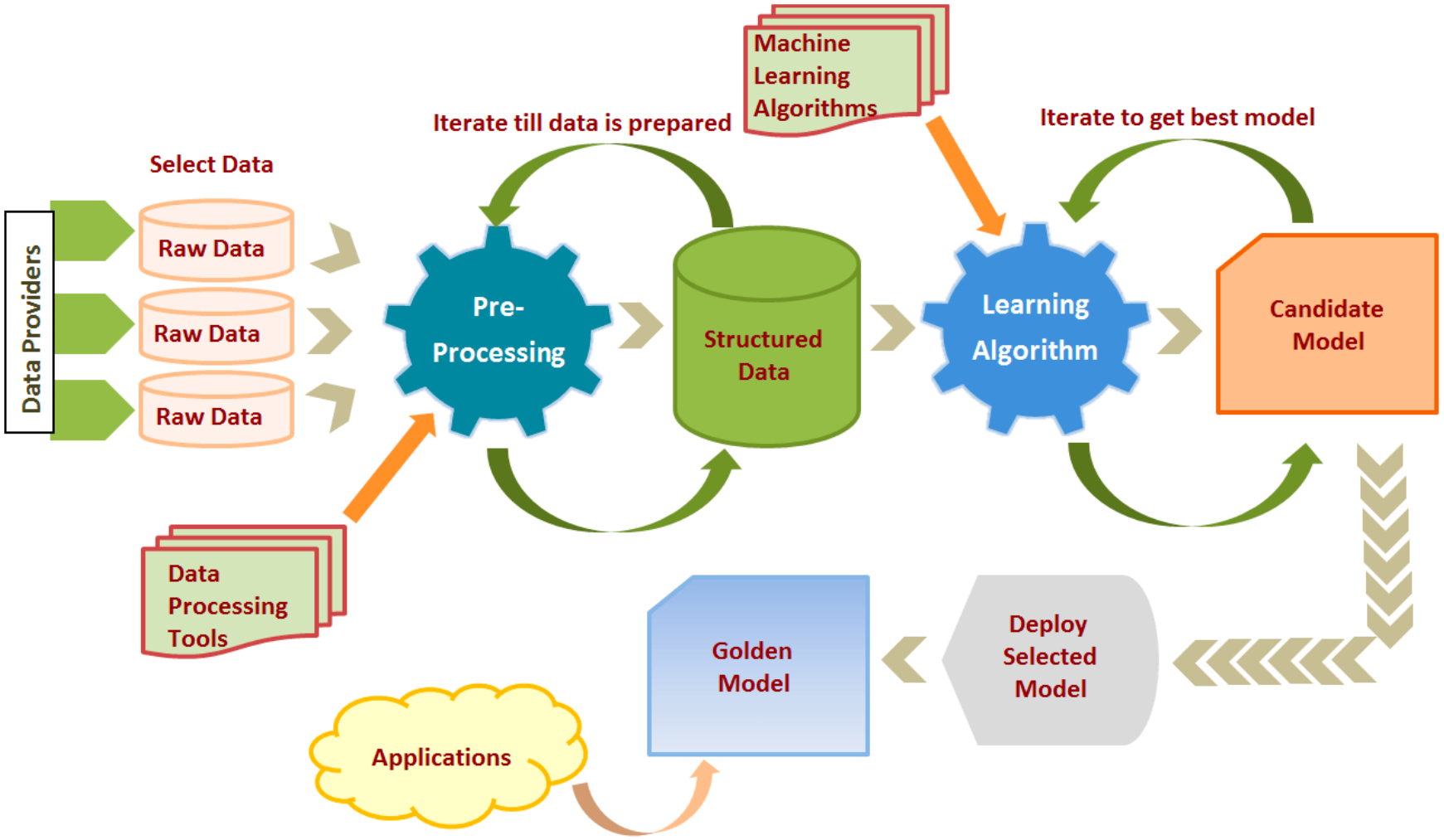

بسم الله الرحمن الرحيم

Fruit of the Slide



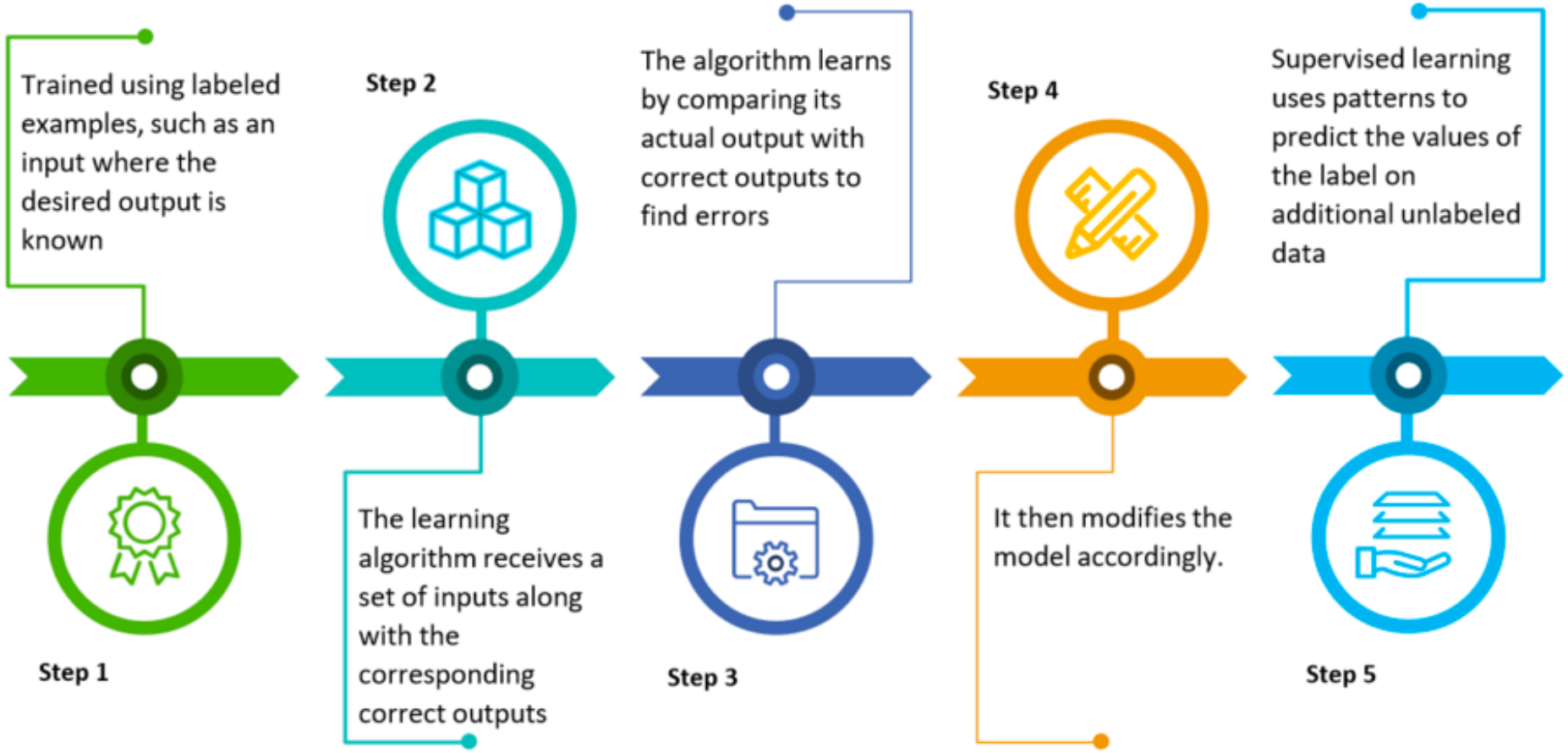
Instructor: Khayyam Salehi, Ph.D.

Machine Learning Scenarios



https://elearningindustry.com/machine-learning-process-and-scenarios

Learning Process



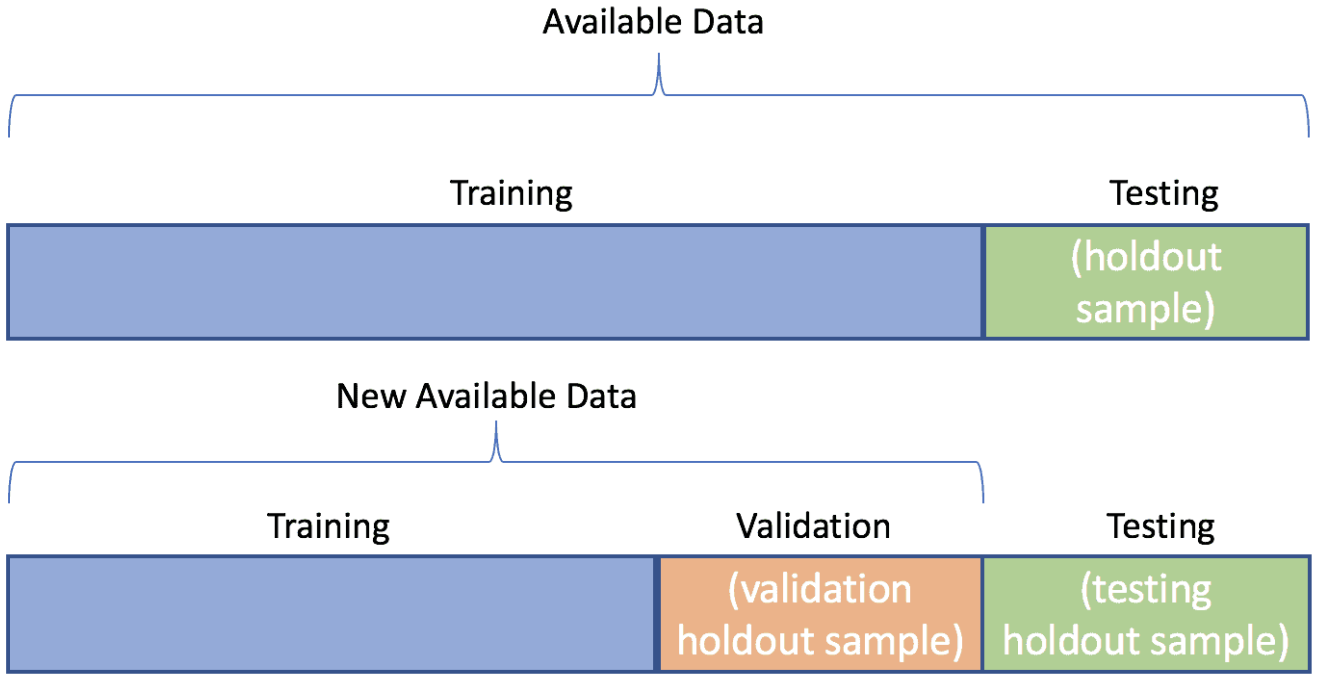
https://www.clariba.com/machine-learning-for-business

Machine Learning Principles

- Using gathered data can predict unseen data.
- Unseen data are the same distribution as the gathered data.

Machine Learning Principles

- Using gathered data can predict unseen data.
- Unseen data are the same distribution as the gathered data.
  - We should testing our model. --> Splitting data into training and testing data.

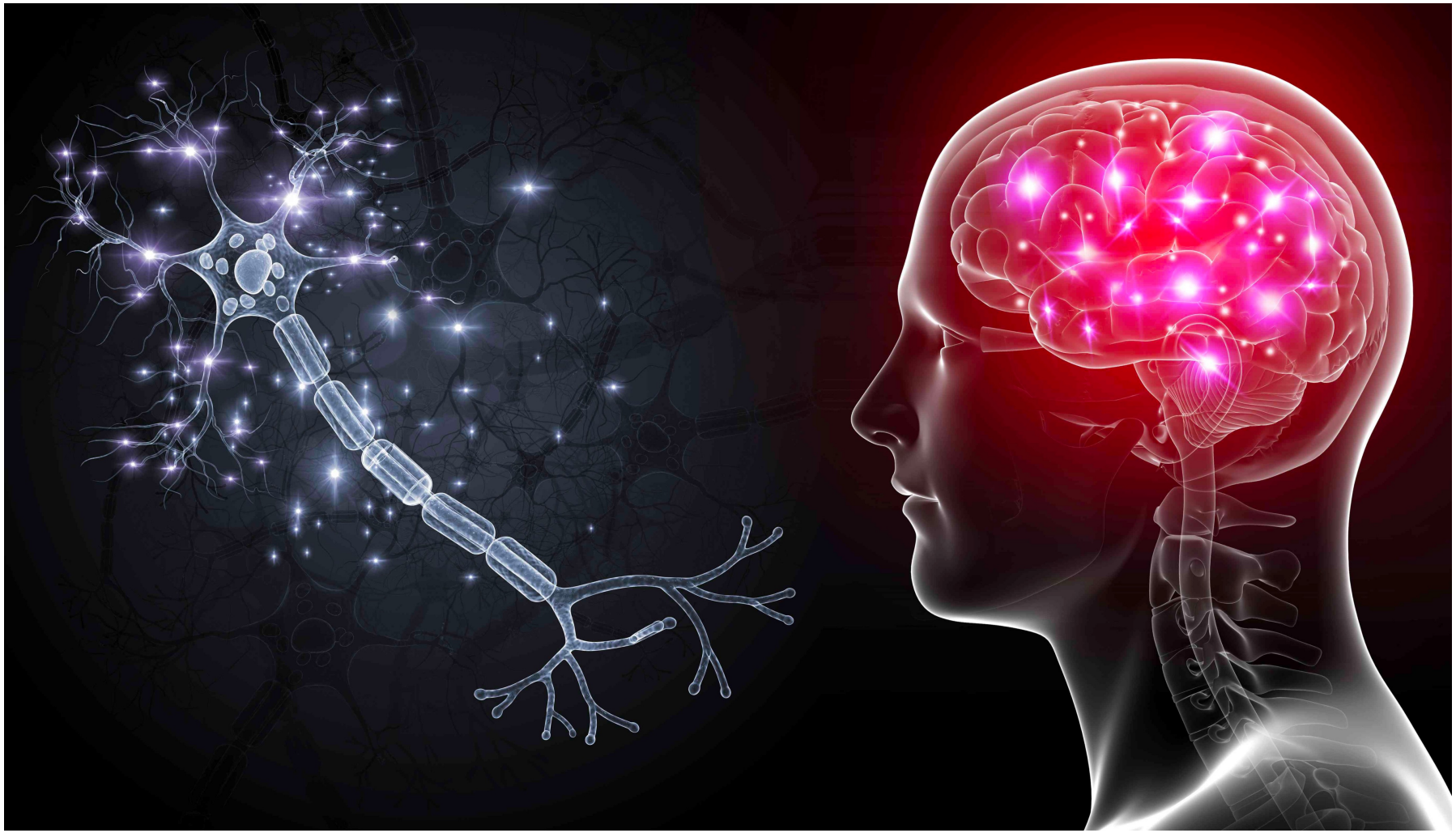


https://algotrading101.com/learn/train-test-split/

Machine Learning Principles

- Using gathered data can predict unseen data.
- Unseen data are the same distribution as the gathered data.
  - We should testing our model. --> Splitting data into training and testing data.
- We have mistakes! --> Accuracy, Precision, Recall ...
- The simpler model, the better model.

How brain cells communicate with each other?

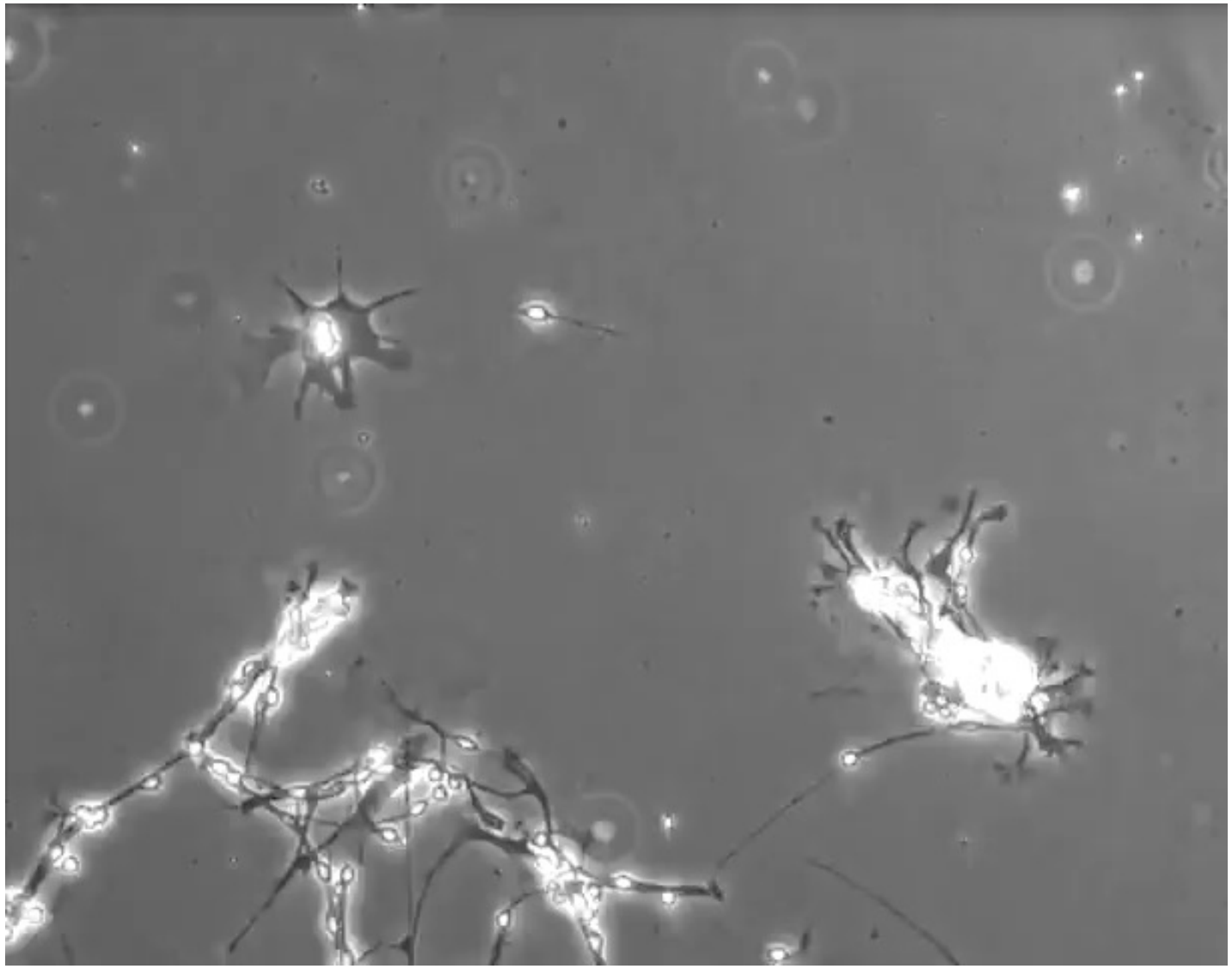


https://www.verywellmind.com/how-brain-cells-communicate-with-each-other-2584397

Single brain cell looking for connections:

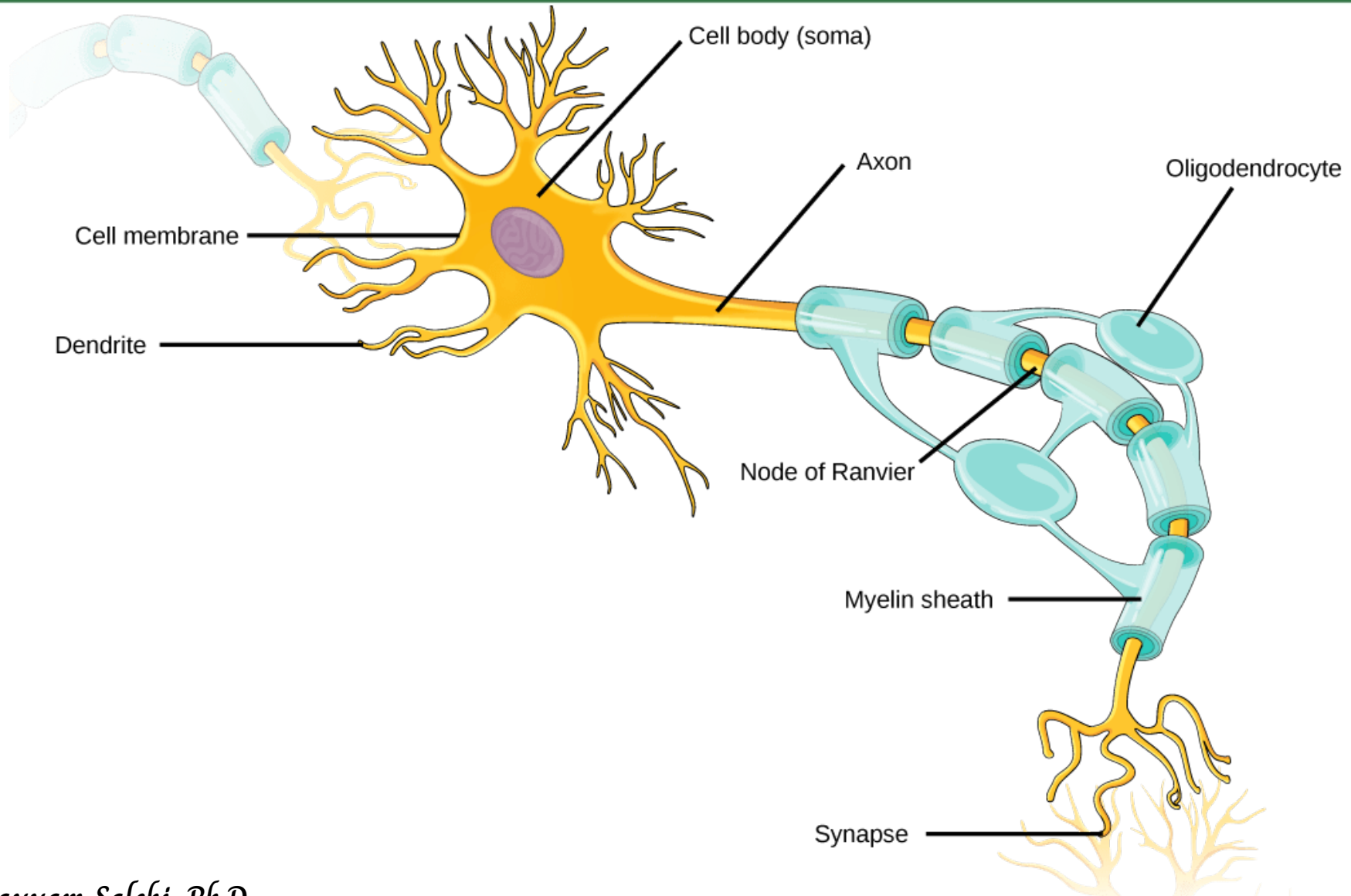
https://www.linkedin.com/posts/slava-bobrov_biology-neuroscience-medicine-activity-6988847118174023680-xZW5?utm_source=share&utm_medium=member_desktop

Neurons forming connections



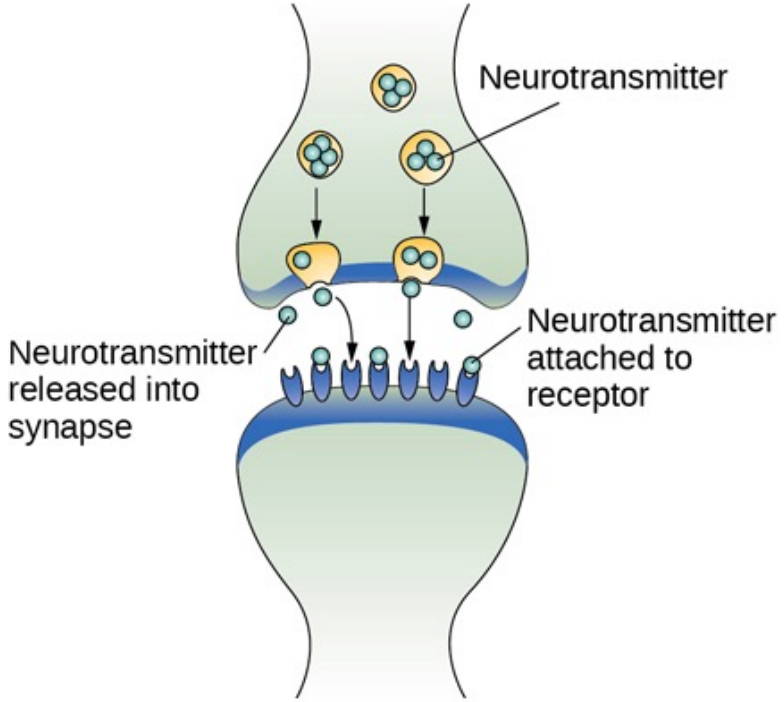
https://www.linkedin.com/posts/slava-bobrov_neuroscience-activity-6856189155949465601-GOHH?utm_source=share&utm_medium=member_desktop

Neurons Structure

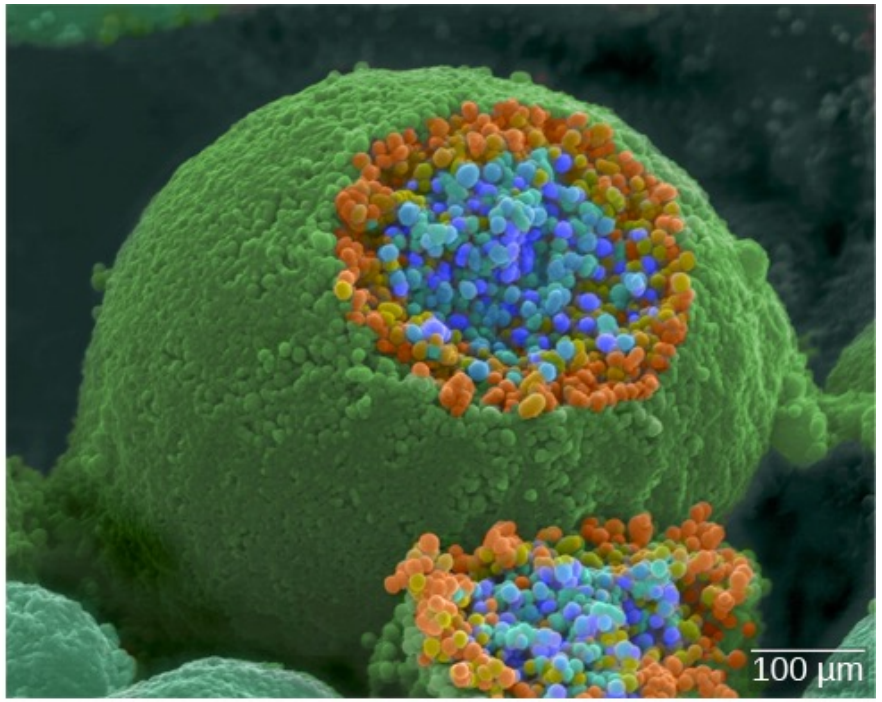


https://organismalbio.biosci.gatech.edu/chemical-and-electrical-signals/neurons/

Communicating



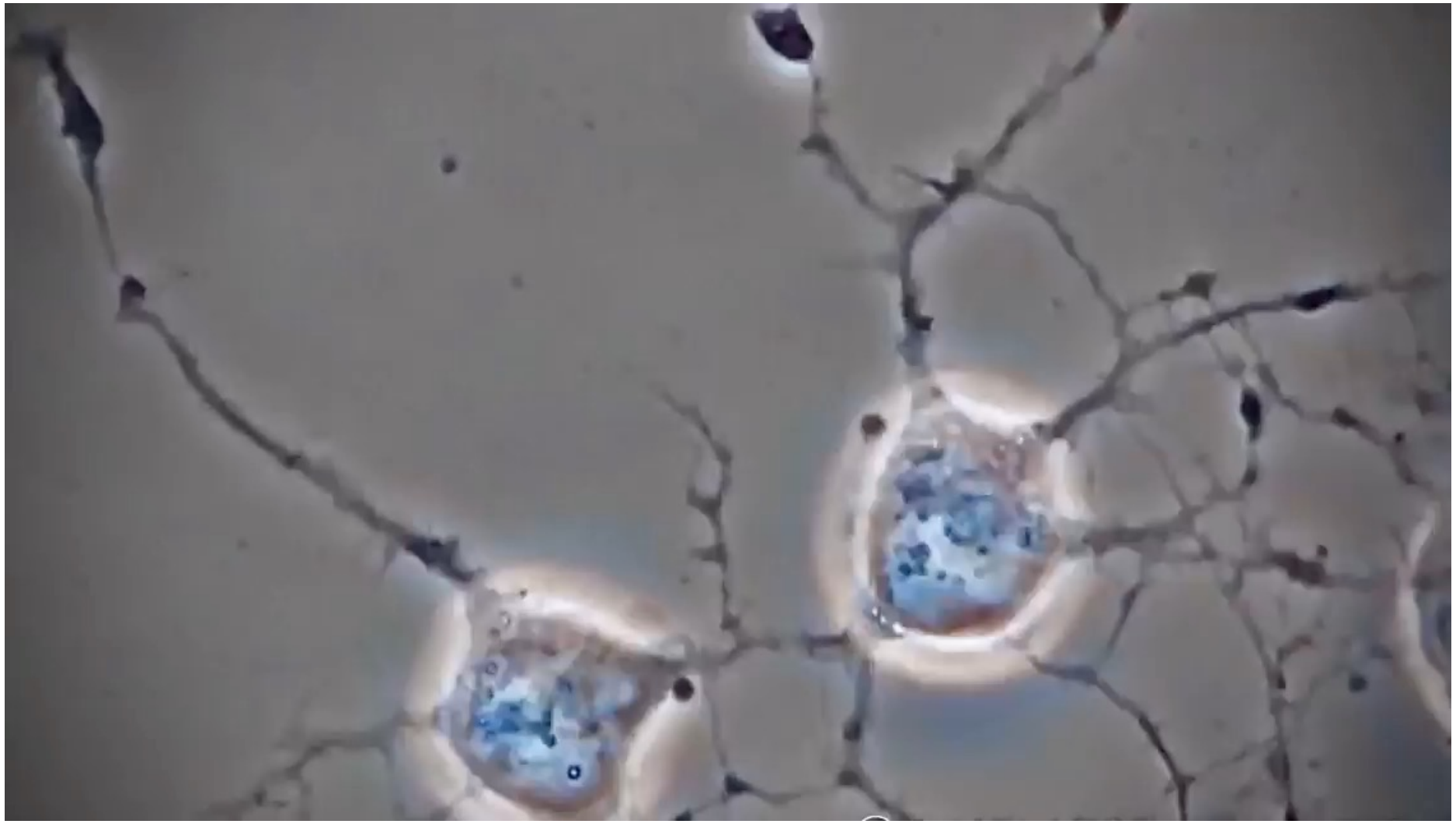
(a)



(b)

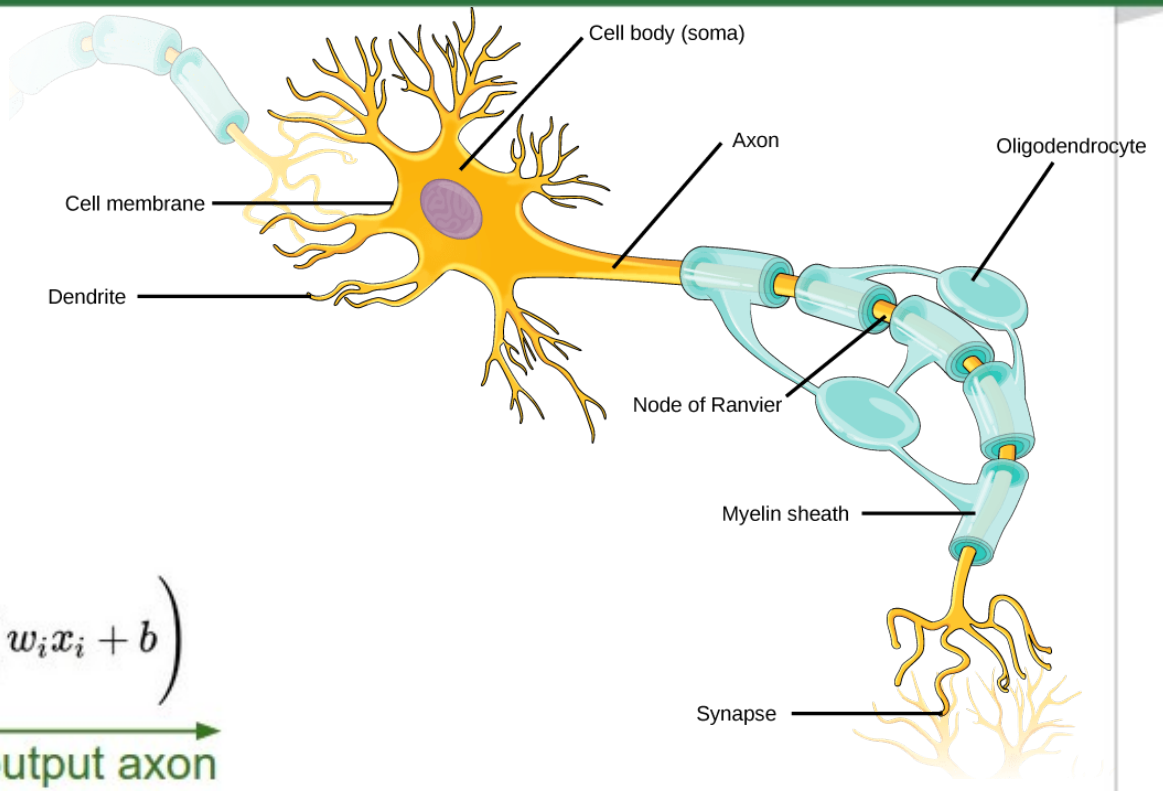
https://organismalbio.biosci.gatech.edu/chemical-and-electrical-signals/neurons/

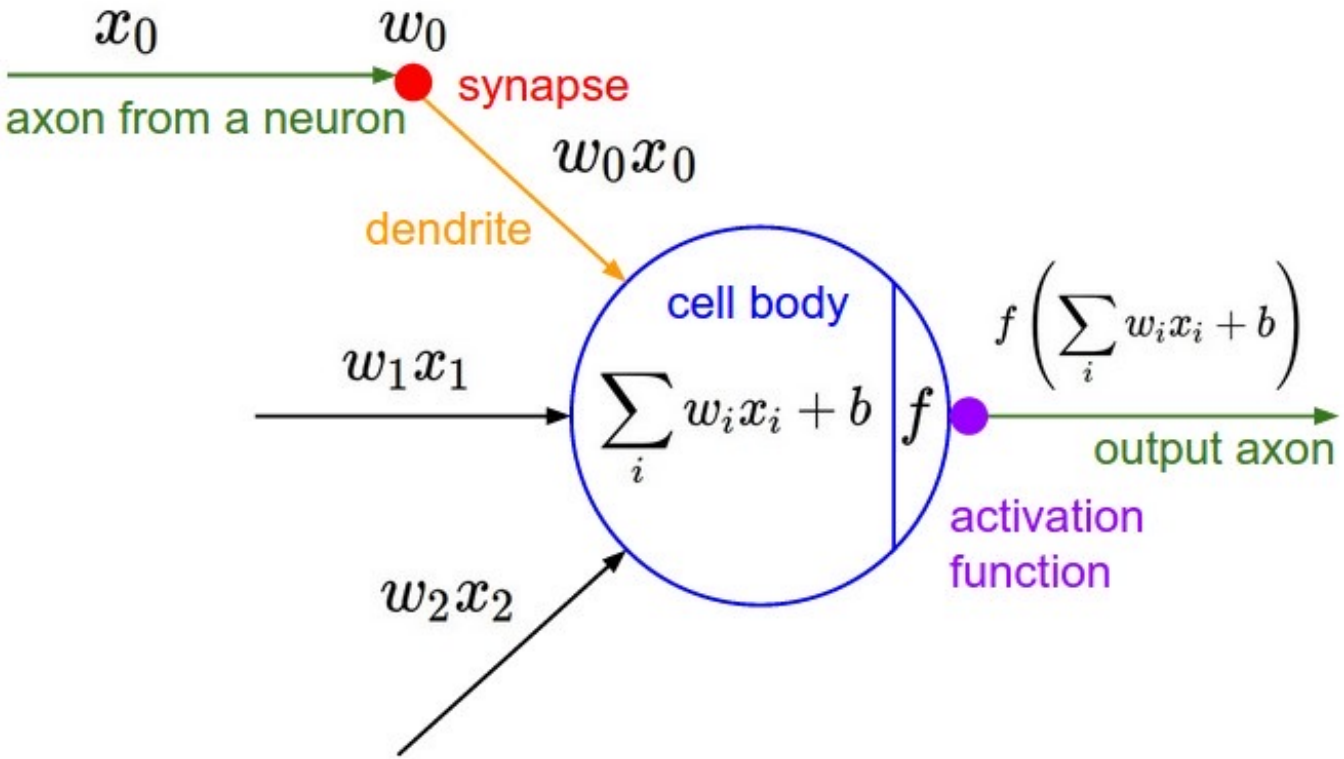
Two brain cells having a chat:



https://www.linkedin.com/posts/slava-bobrov_neuroscience-biology-medicine-activity-7060589883249020928-L06e?utm_source=share&utm_medium=member_desktop

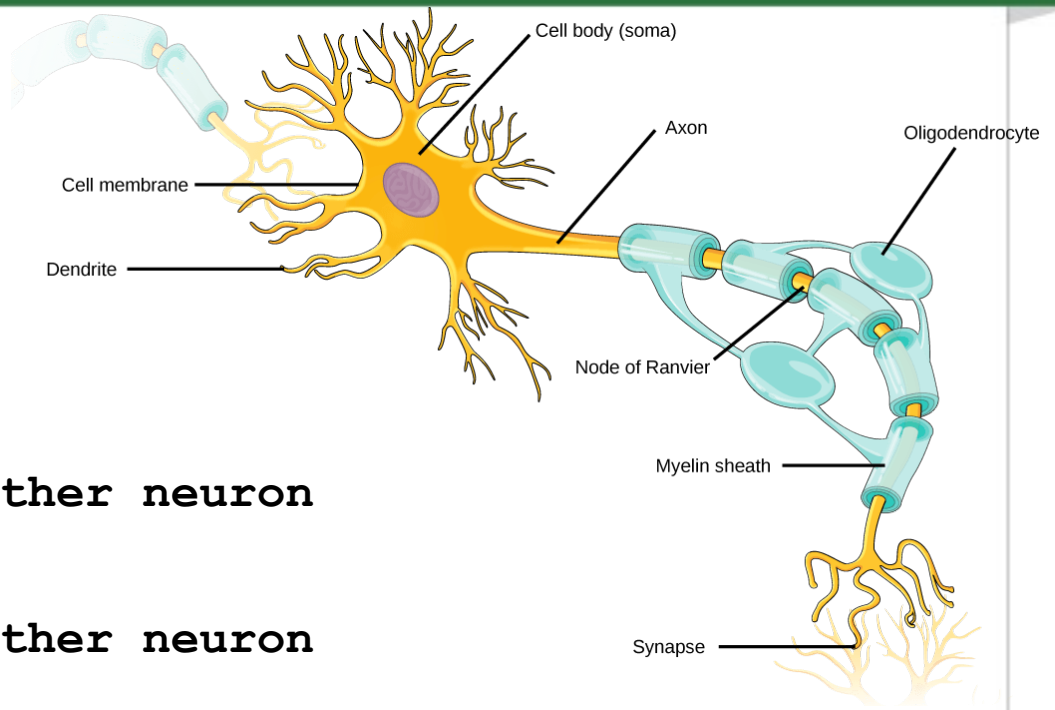
Modeling a Neuron (Perceptron)

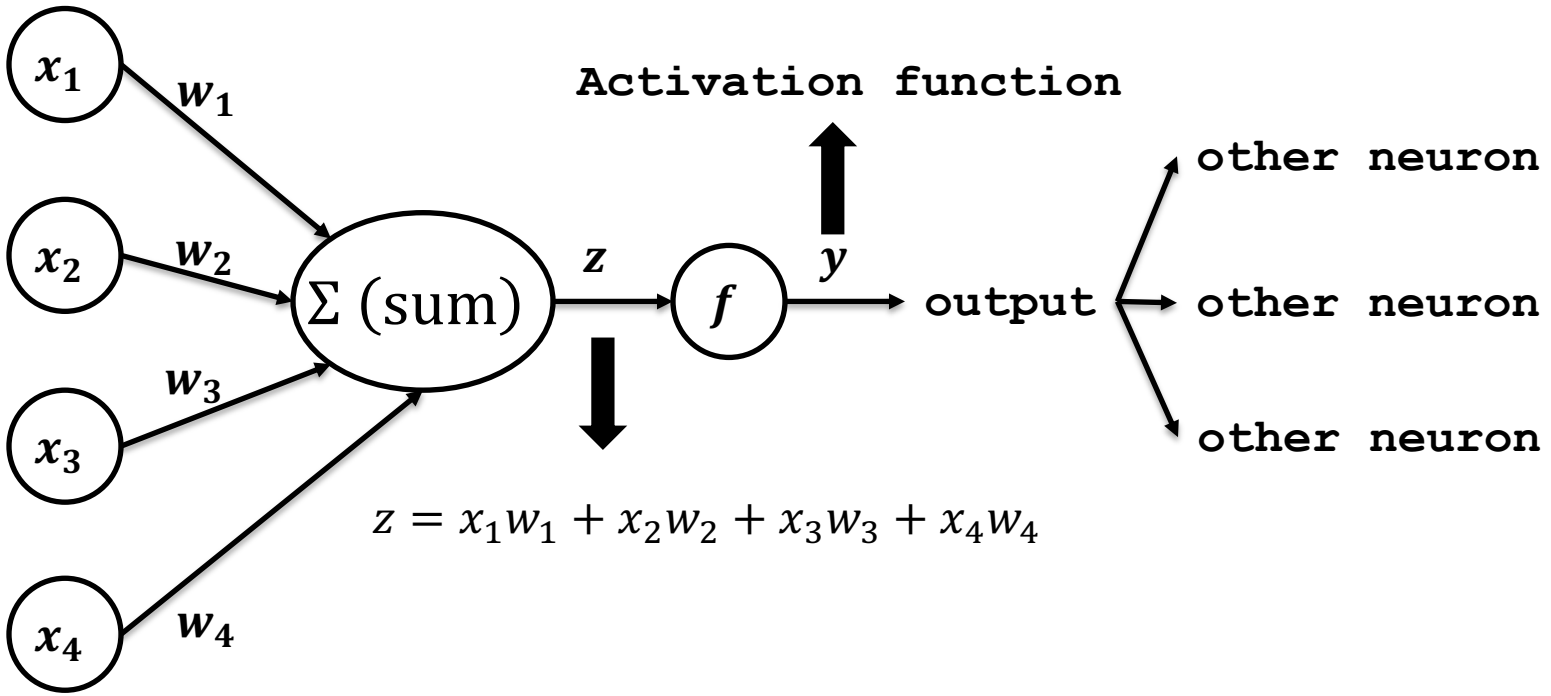




https://leonardoaraujosantos.gitbook.io/artificial-inteligence/machine_learning/supervised_learning/neural_networks

Modeling a Neuron (Perceptron)

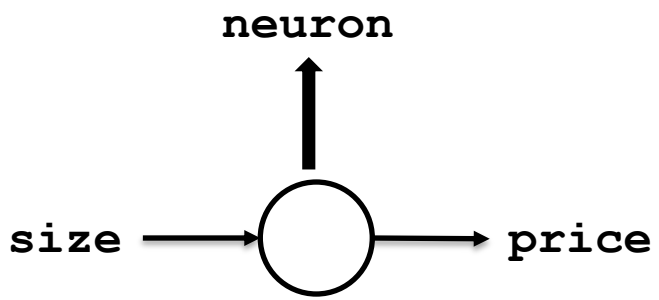




$$z = x_1w_1 + x_2w_2 + x_3w_3 + x_4w_4$$

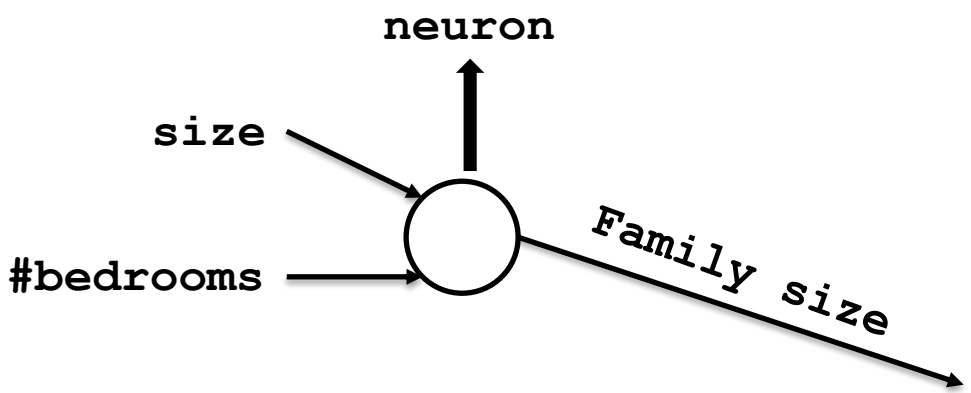
What is a Neural Network?

Predicting a price of house just based on the size (feature):



What is a Neural Network?

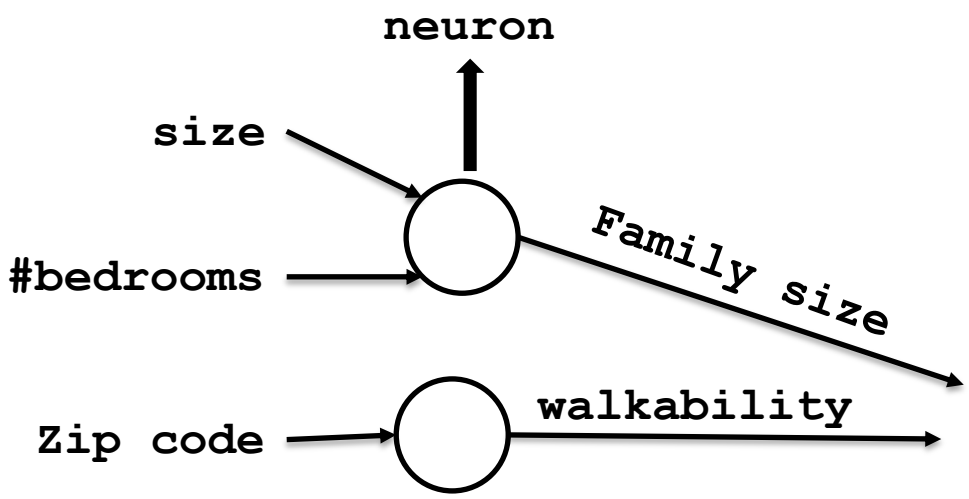
Predicting a price of house just based on some features:



Thanks to Andrew NG, deeplearning.ai

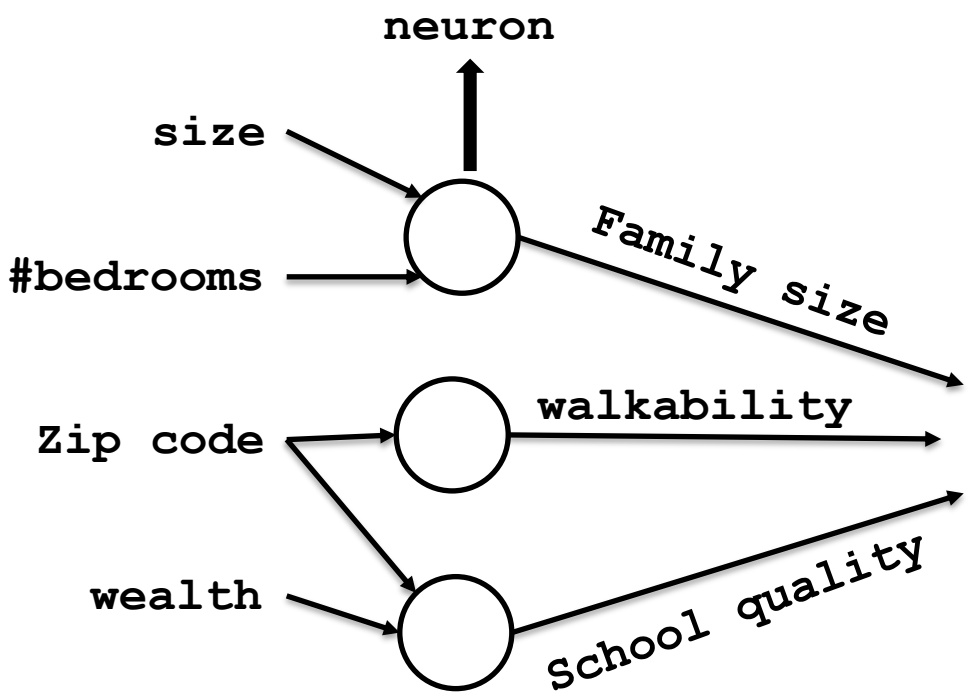
What is a Neural Network?

Predicting a price of house just based on some features:



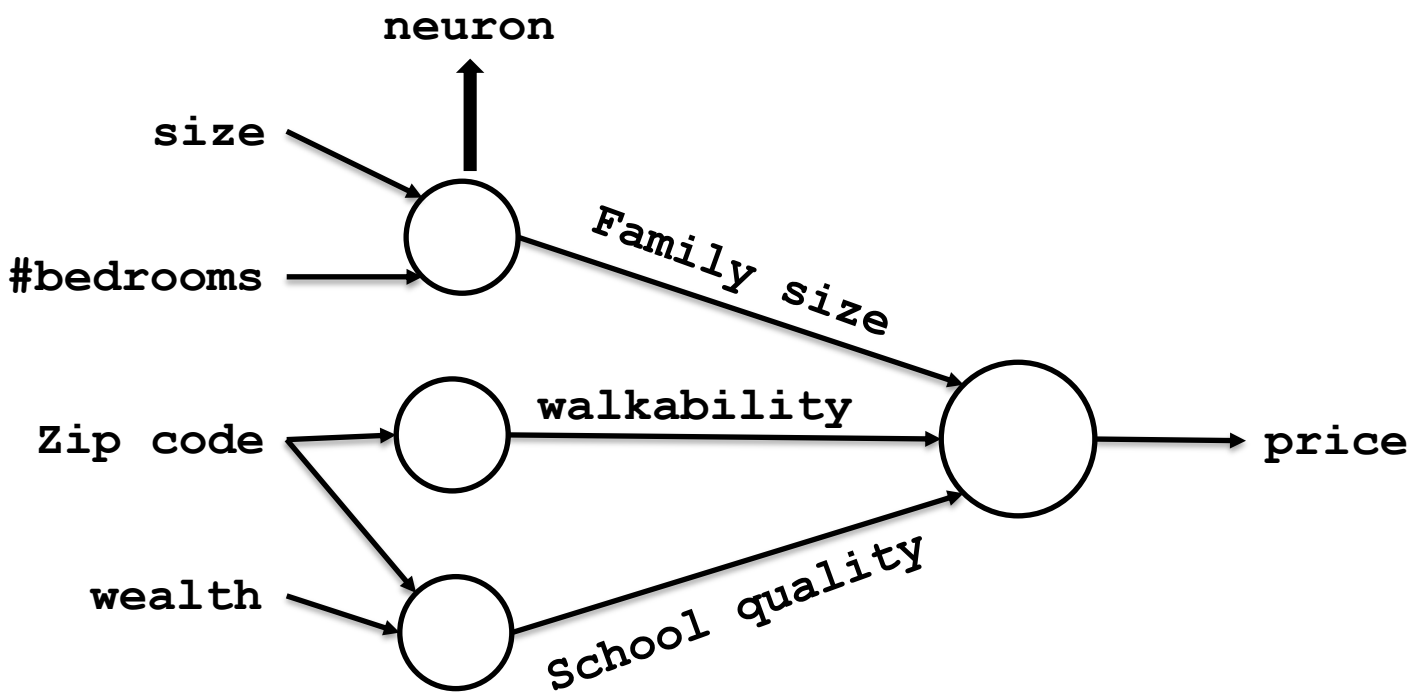
What is a Neural Network?

Predicting a price of house just based on some features:



What is a Neural Network?

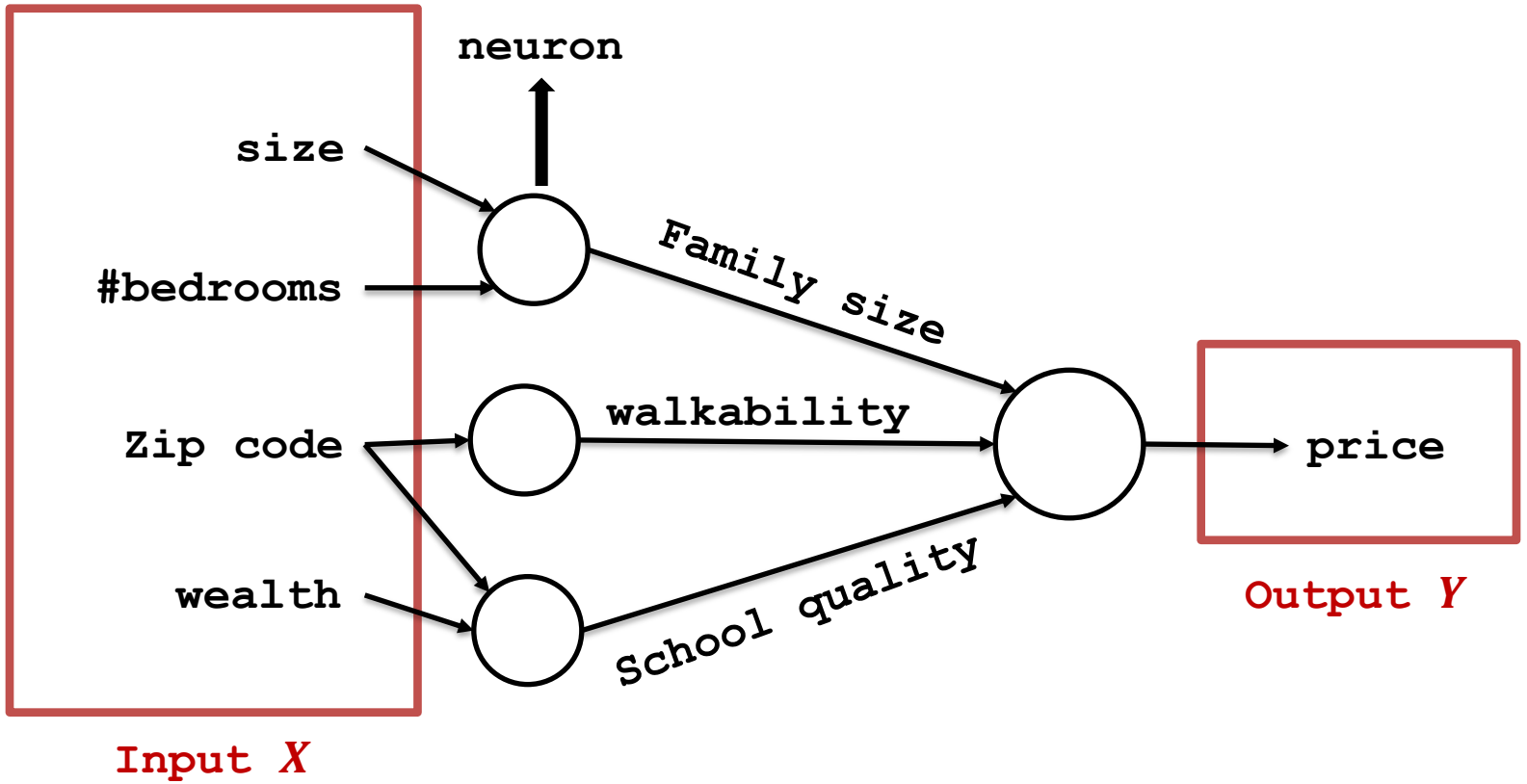
Predicting a price of house just based on some features:



Thanks to Andrew NG, deeplearning.ai

What is a Neural Network?

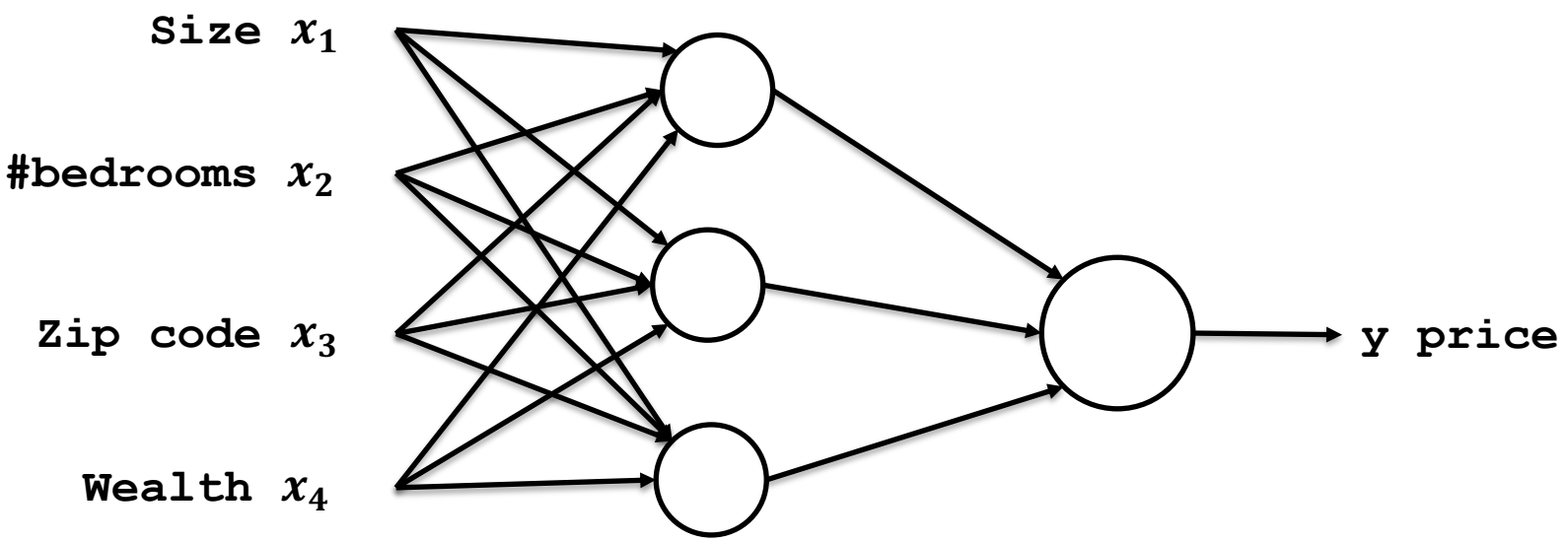
Predicting a price of house just based on some features:



Thanks to Andrew NG, deeplearning.ai

What is a Neural Network?

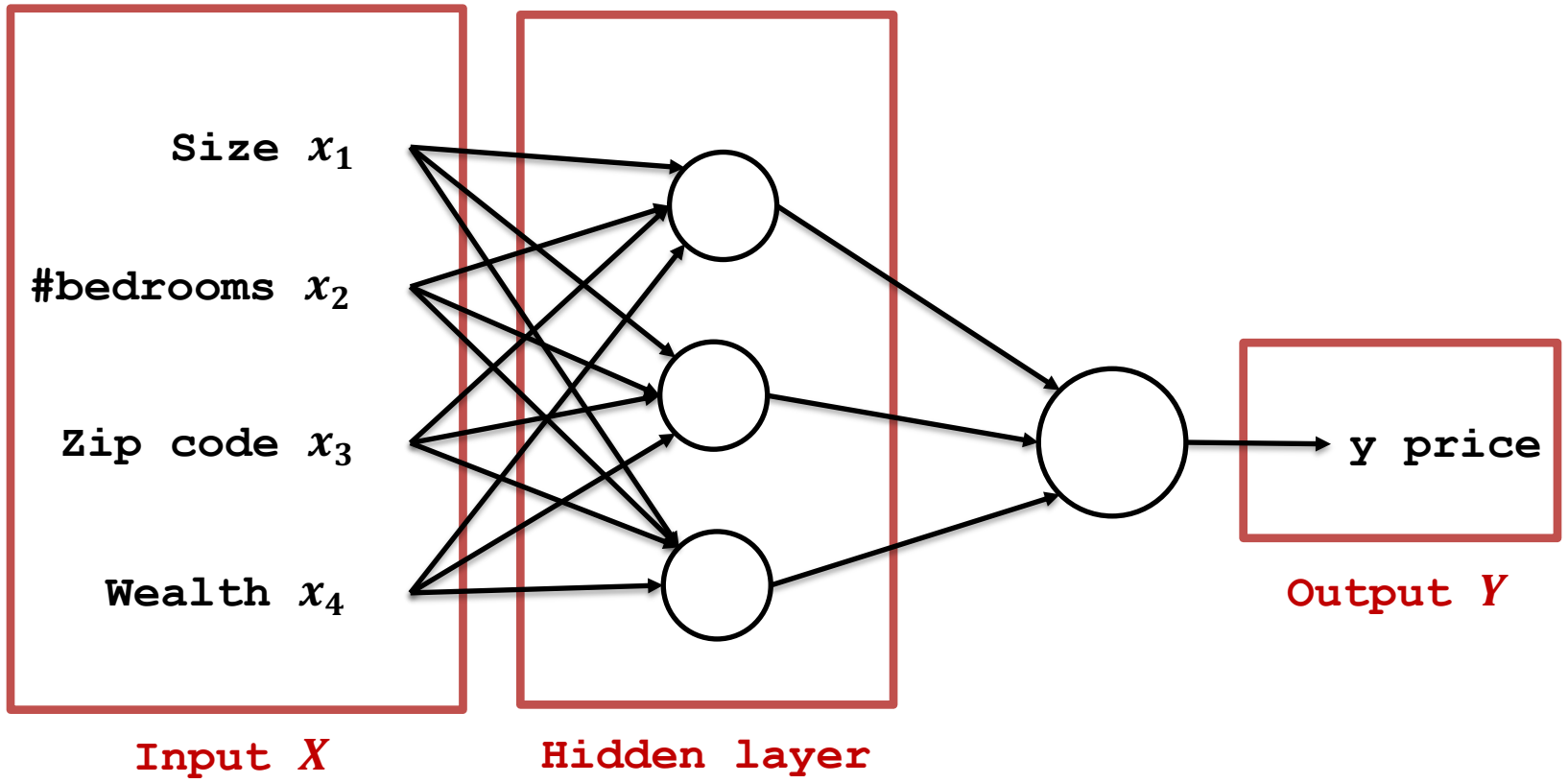
Predicting a price of house just based on some features:



Thanks to Andrew NG, deeplearning.ai

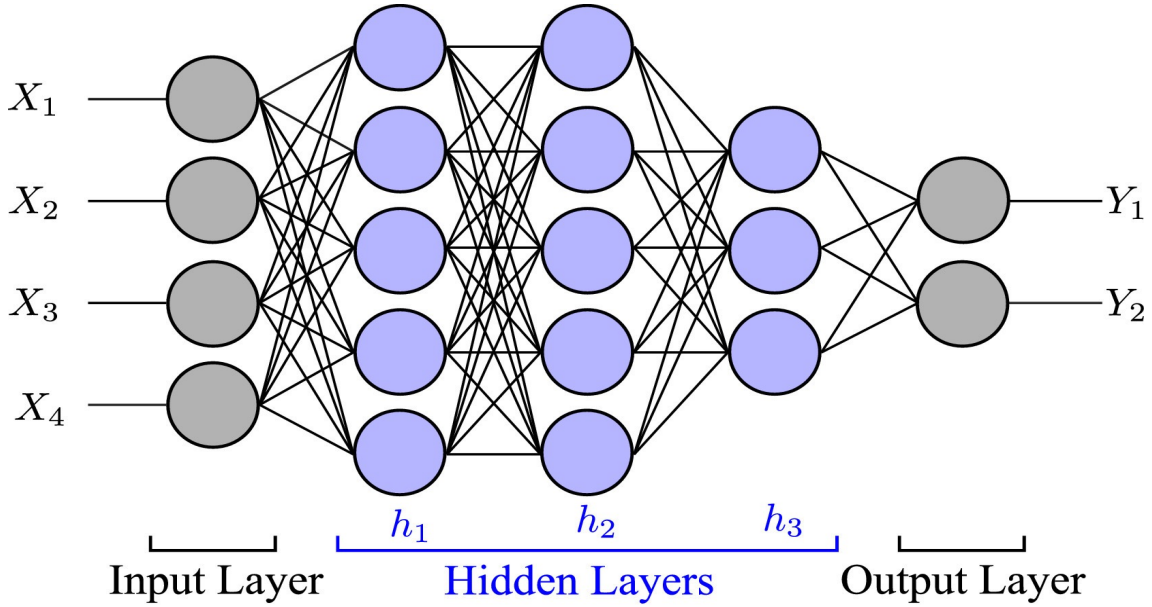
Multi Layer Perceptron (MLP)

Predicting a price of house just based on some features:

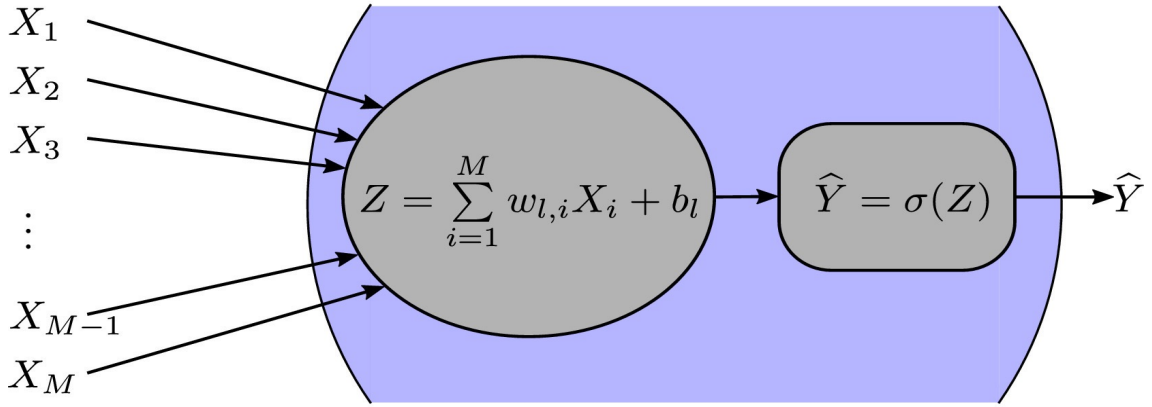


Thanks to Andrew NG, deeplearning.ai

Schematic of a fully connected neural network



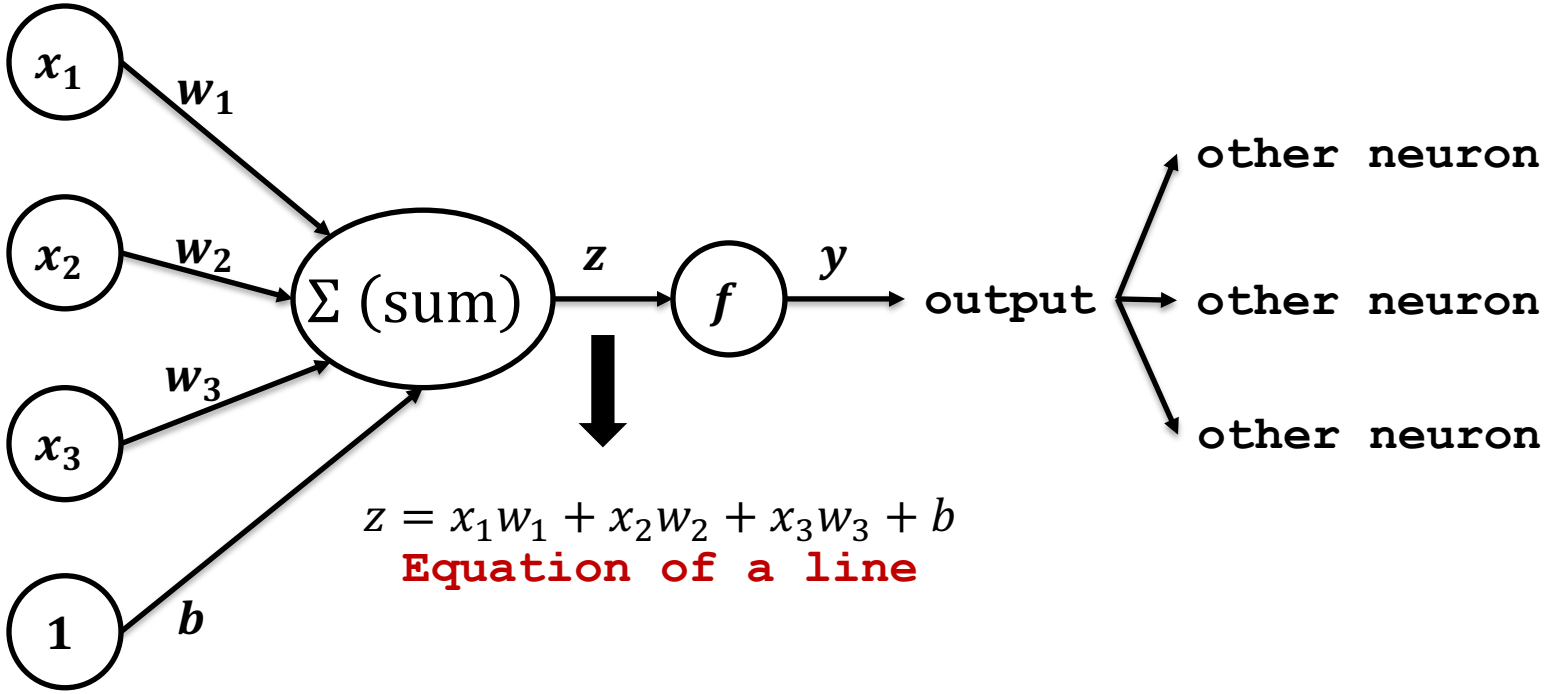
(a) Network architecture.



(b) Representation of a neuron.

https://www.sciencedirect.com/science/article/pii/S0360128522000193

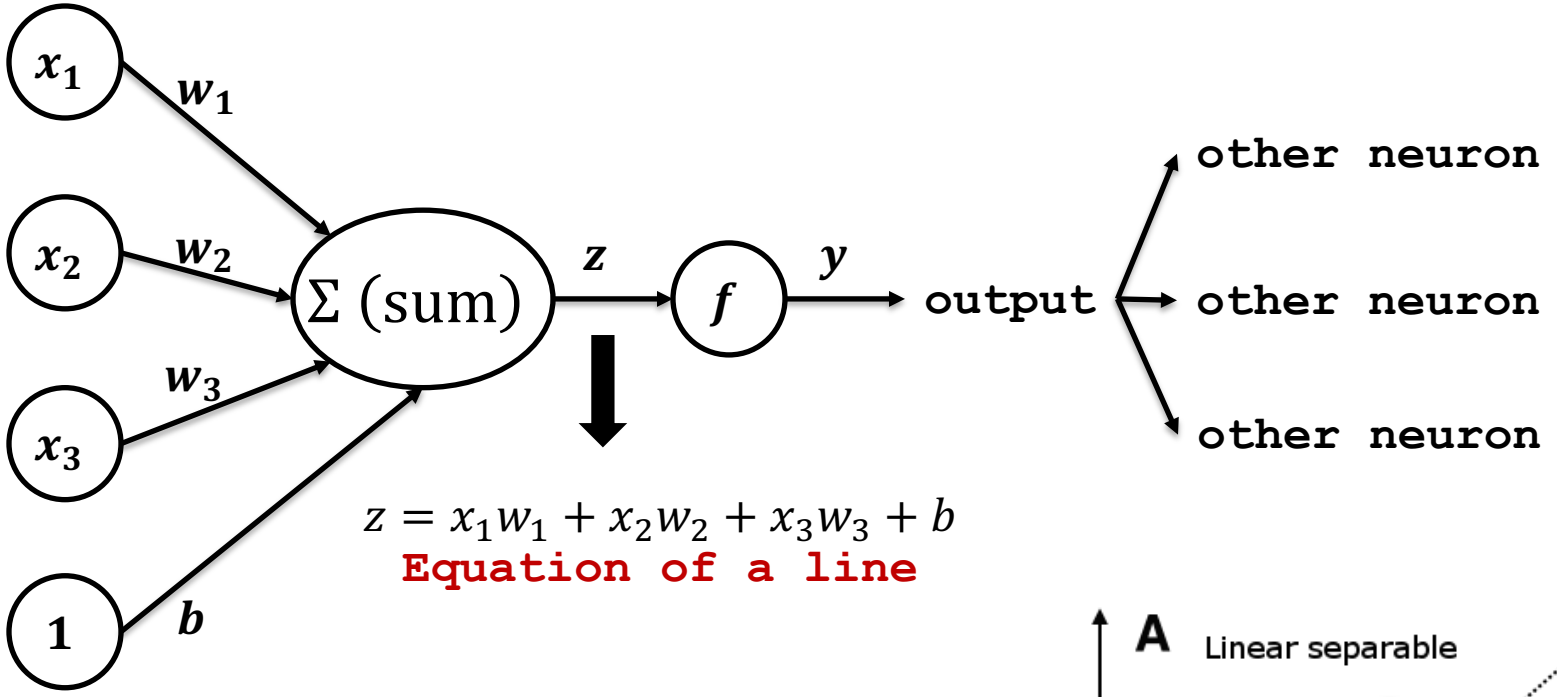
The need for MLPs



$$z = x_1w_1 + x_2w_2 + x_3w_3 + b$$

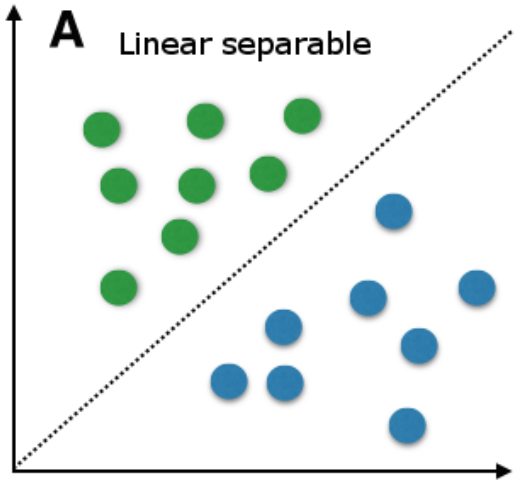
Equation of a line

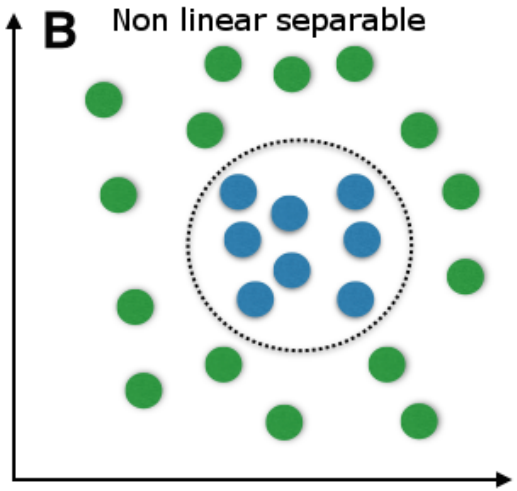
The need for MLPs



$$z = x_1 w_1 + x_2 w_2 + x_3 w_3 + b$$

Equation of a line





https://leonardoaraujosantos.gitbook.io/artificial-inteligence/machine_learning/supervised_learning/linear_classification

XOR Problem

AND Truth Table

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

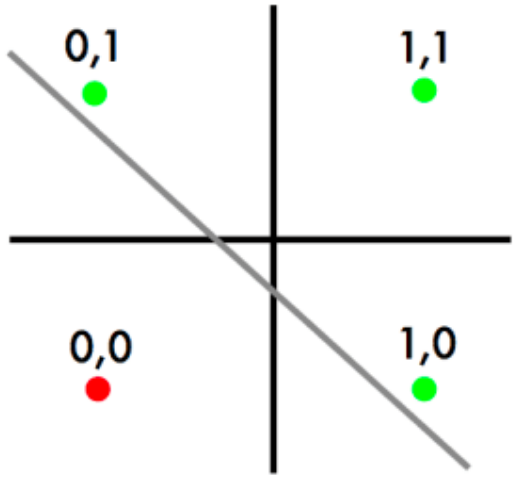
XOR Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

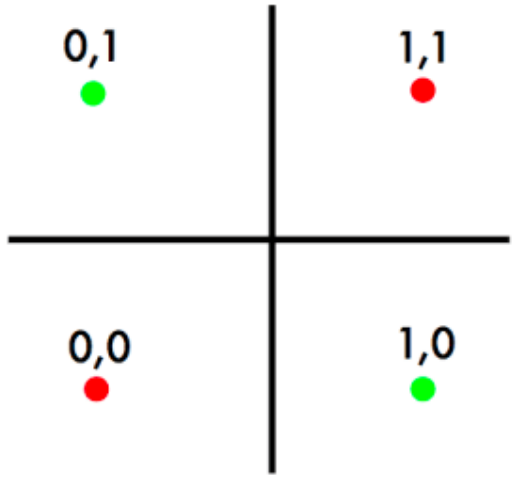
NOT Truth Tabl

A	B
0	1
1	0

The XOR problem



OR



XOR

Forward Propagation

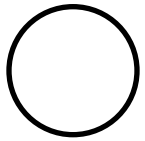
	Feature 1	Feature 2
Samples	0.5	-0.3
	0.4	0.1
	0.6	0.8

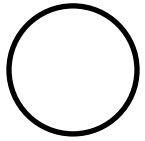
Output 1	Output 2
0.9	0.1
0.9	0.8
0.1	0.1

Forward Propagation

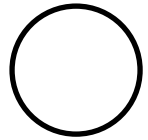
	Feature 1	Feature 2
Samples	0.5	-0.3
	0.4	0.1
	0.6	0.8

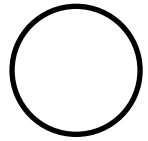
Output 1	Output 2
0.9	0.1
0.9	0.8
0.1	0.1





Input Layer



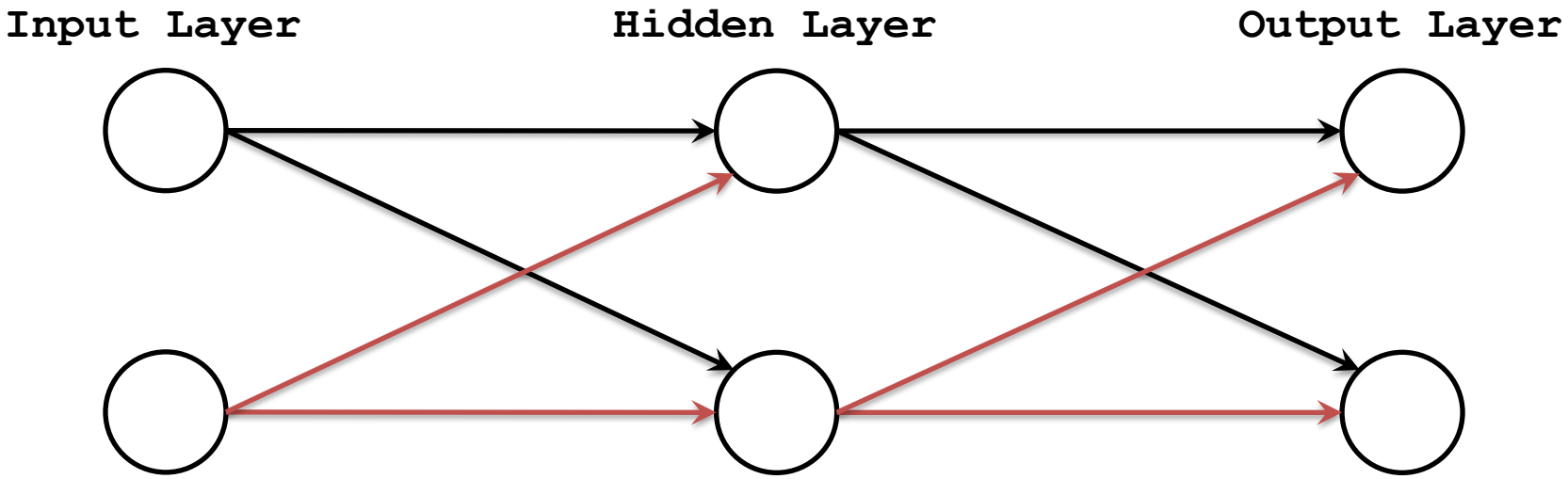


Output Layer

Forward Propagation

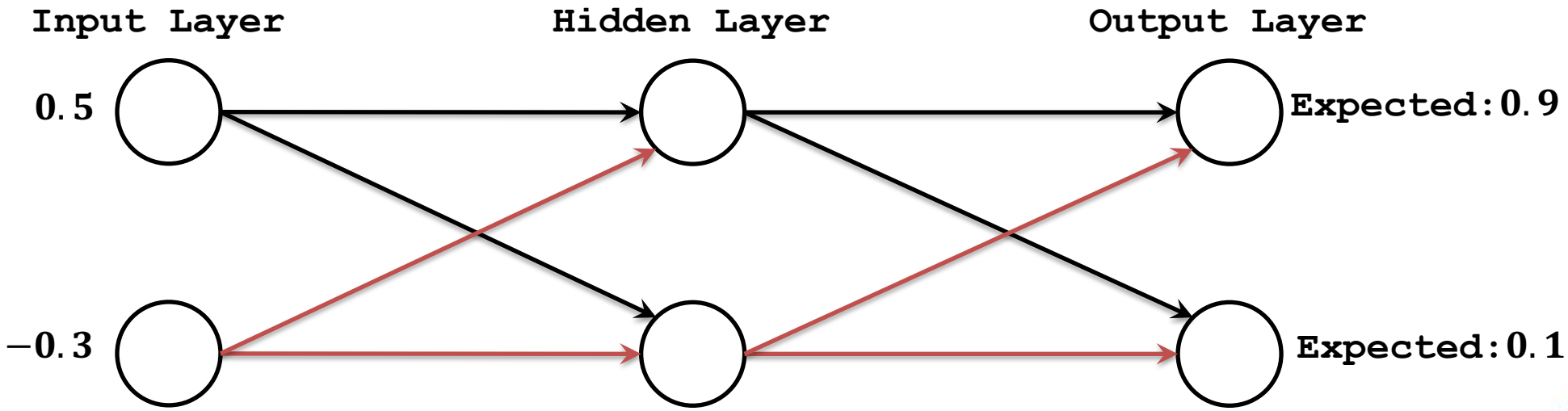
	Feature 1	Feature 2
Samples	0.5	-0.3
	0.4	0.1
	0.6	0.8

Output 1	Output 2
0.9	0.1
0.9	0.8
0.1	0.1



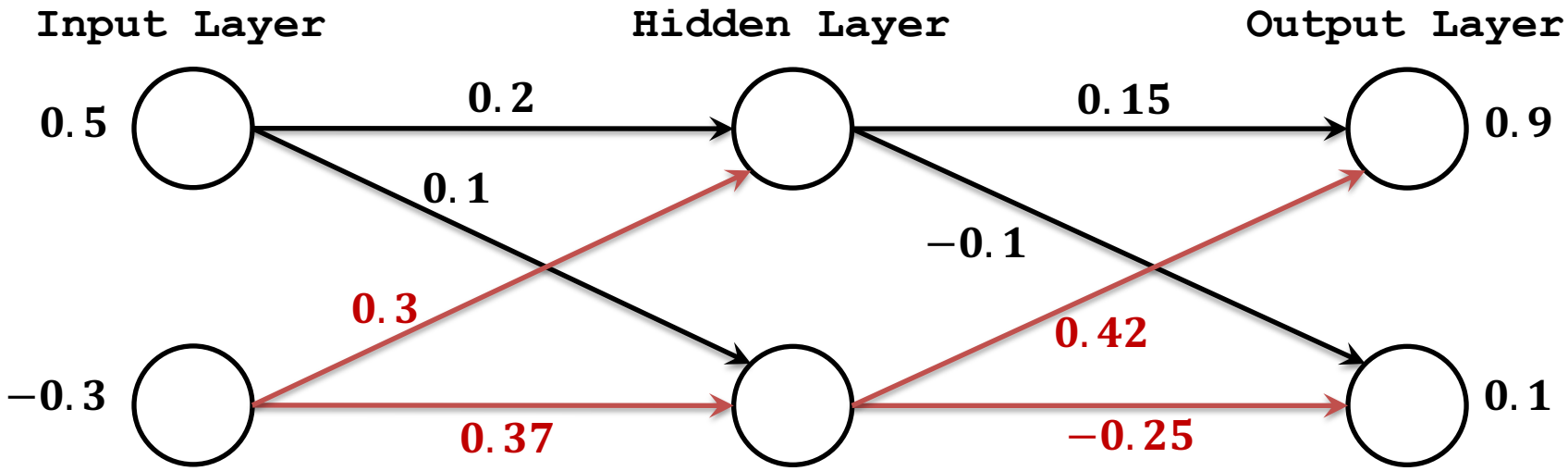
Forward Propagation

	Feature 1	Feature 2	Output 1	Output 2
Samples	0.5	-0.3	0.9	0.1
	0.4	0.1	0.9	0.8
	0.6	0.8	0.1	0.1



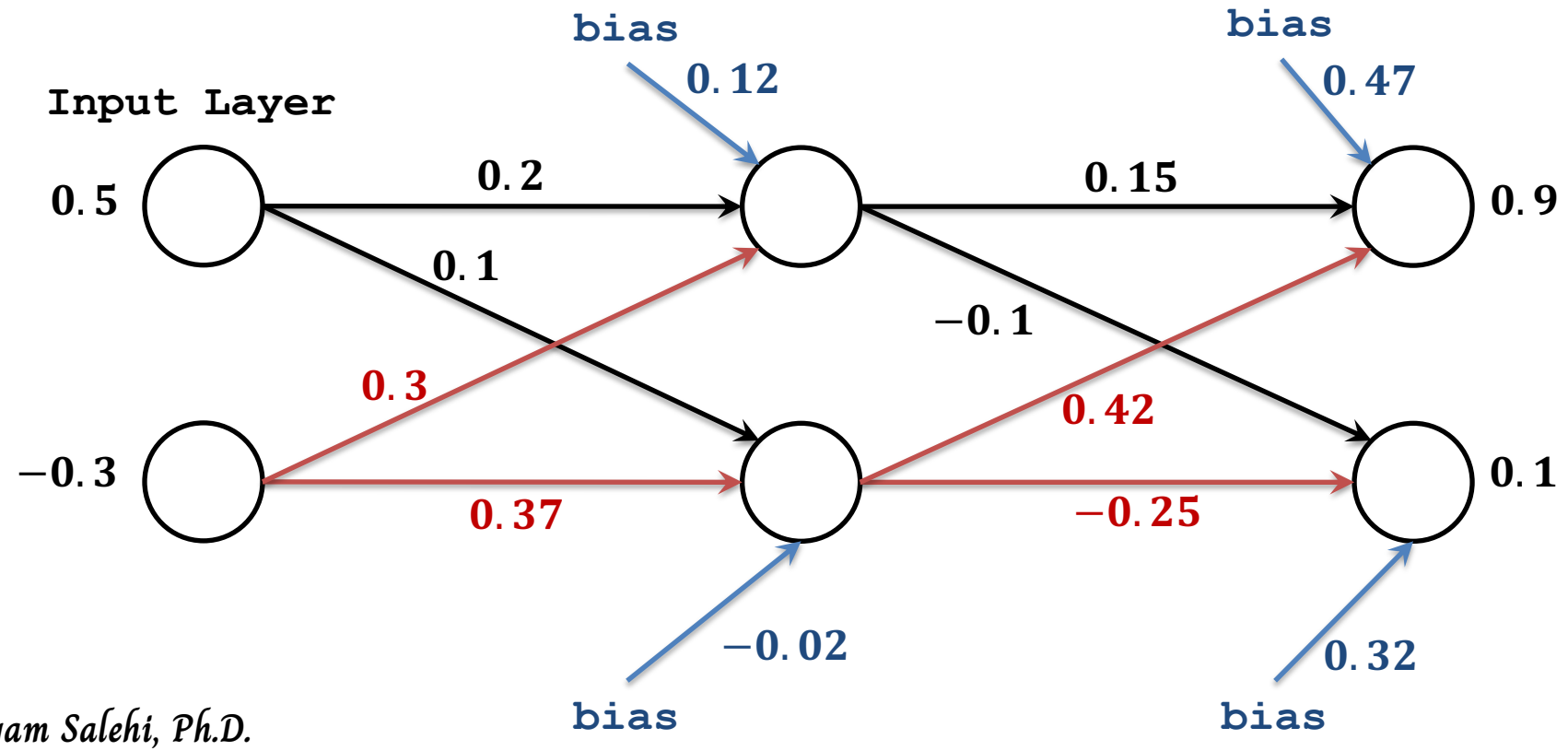
Forward Propagation

	Feature 1	Feature 2	Output 1	Output 2
Samples	0.5	-0.3	0.9	0.1
	0.4	0.1	0.9	0.8
	0.6	0.8	0.1	0.1



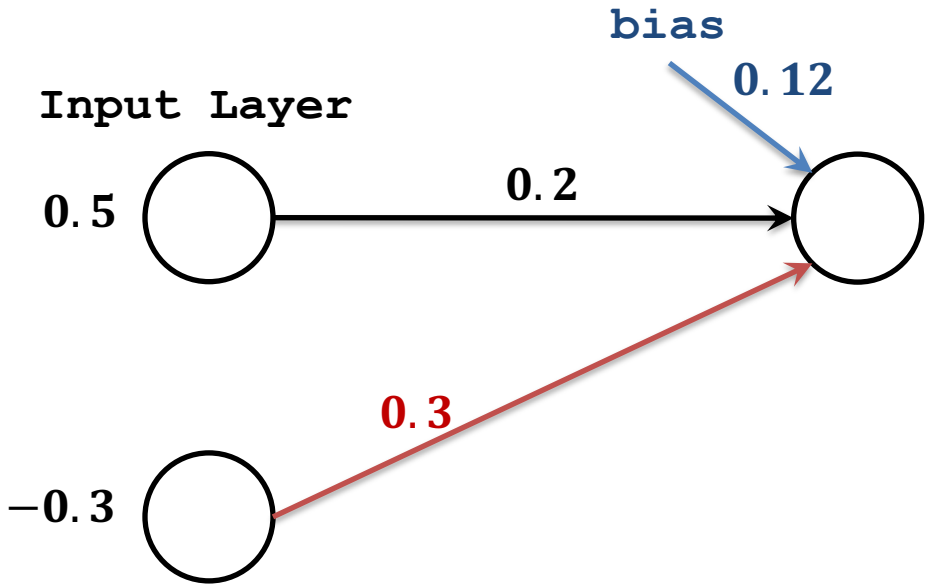
Forward Propagation

	Feature 1	Feature 2	Output 1	Output 2
Samples	0.5	-0.3	0.9	0.1
	0.4	0.1	0.9	0.8
	0.6	0.8	0.1	0.1



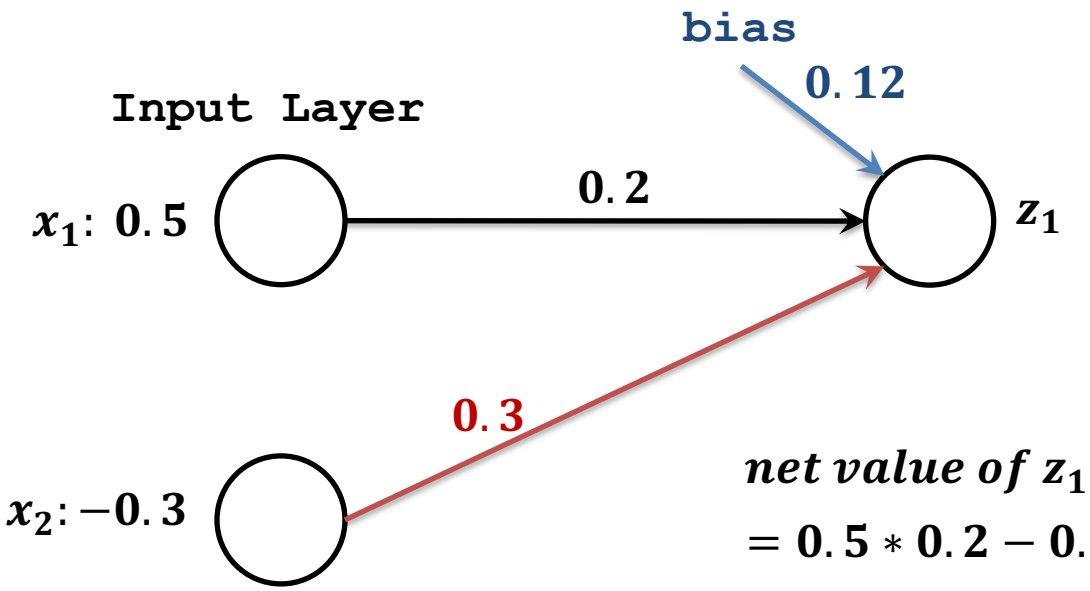
Forward Propagation

	Feature 1	Feature 2	Output 1	Output 2
Samples	0.5	-0.3	0.9	0.1
	0.4	0.1	0.9	0.8
	0.6	0.8	0.1	0.1



Forward Propagation

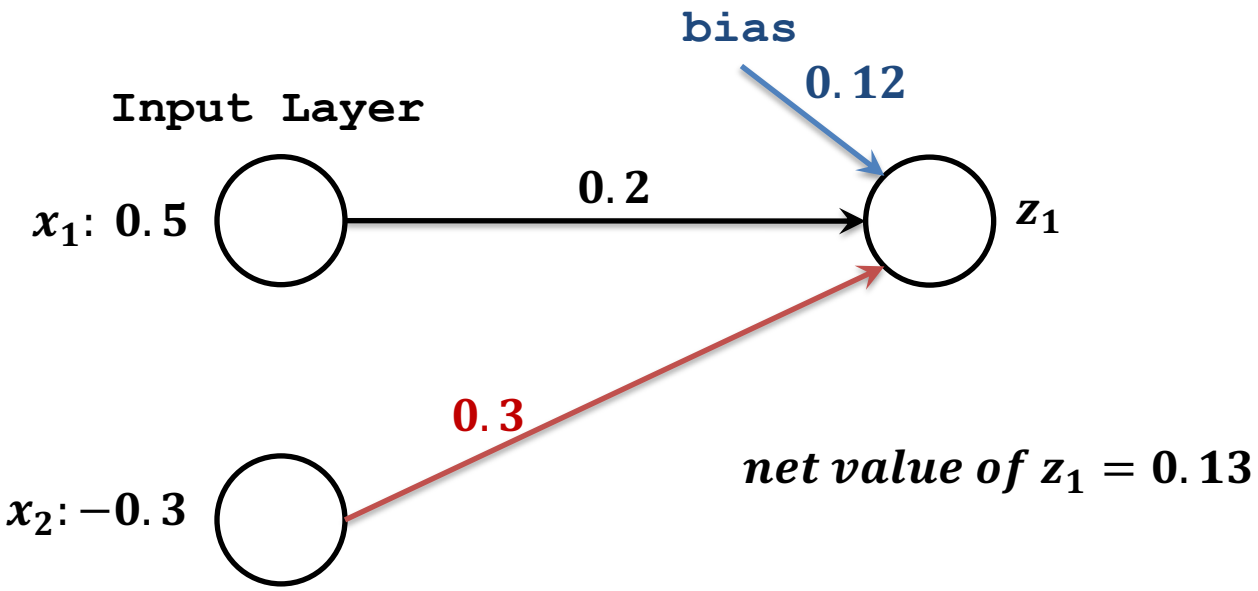
	Feature 1	Feature 2	Output 1	Output 2
Samples	0.5	-0.3	0.9	0.1
	0.4	0.1	0.9	0.8
	0.6	0.8	0.1	0.1

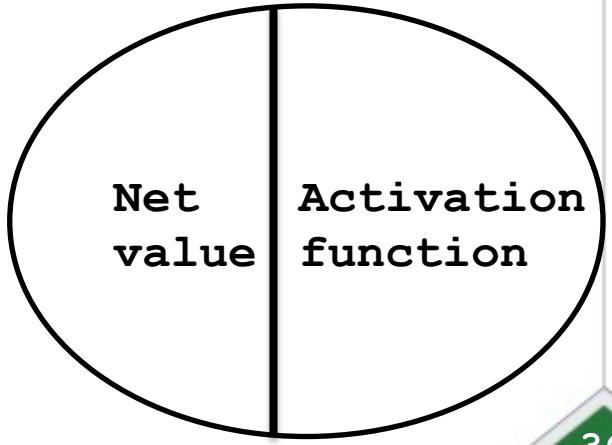


$$net\ value\ of\ z_1 = x_1 w_1 + x_2 w_2 + bias$$
$$= 0.5 * 0.2 - 0.3 * 0.3 + 0.12 = 0.13$$

Forward Propagation

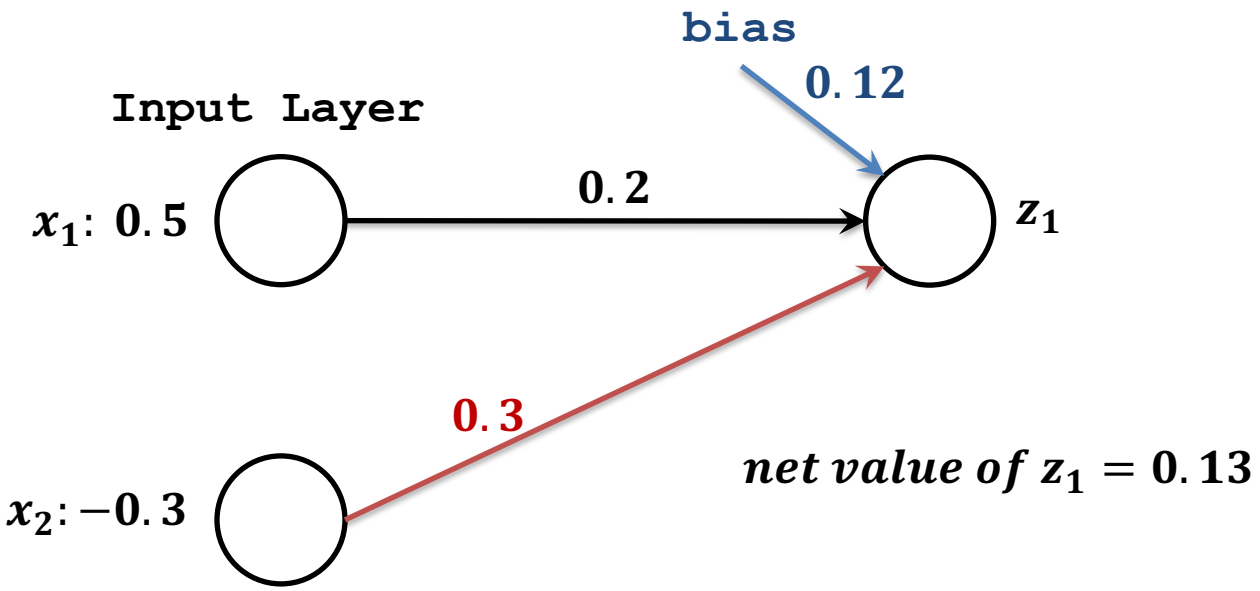
	Feature 1	Feature 2	Output 1	Output 2
Samples	0.5	-0.3	0.9	0.1
	0.4	0.1	0.9	0.8
	0.6	0.8	0.1	0.1

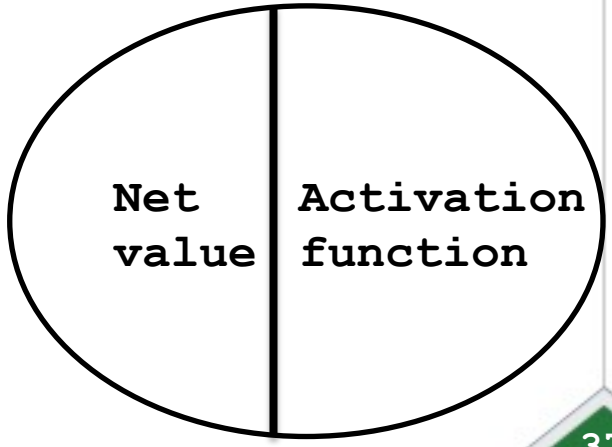




Forward Propagation

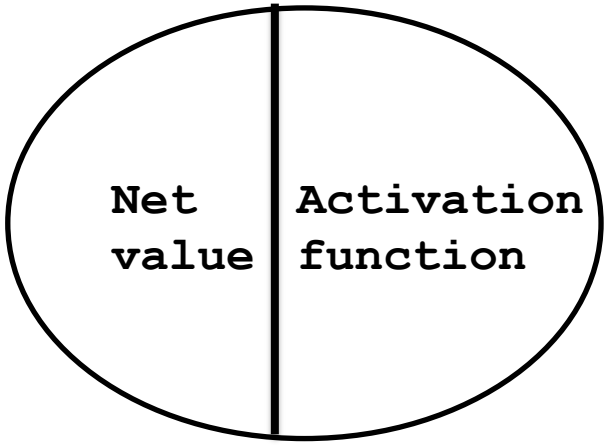
	Feature 1	Feature 2	Output 1	Output 2
Samples	0.5	-0.3	0.9	0.1
	0.4	0.1	0.9	0.8
	0.6	0.8	0.1	0.1



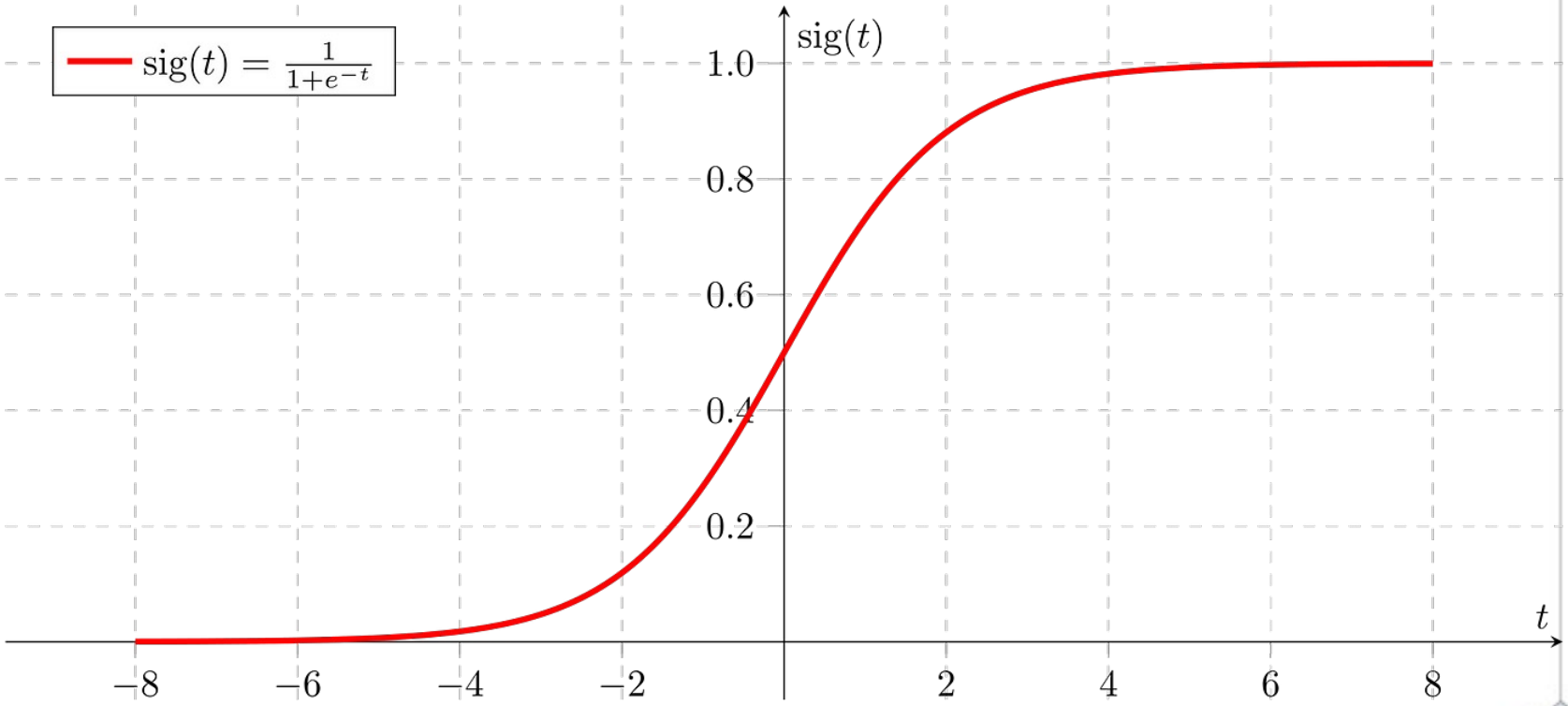


Activation function fires the net value, e.g. scaling it to range $[0,1]$.

A Famous Activation Function

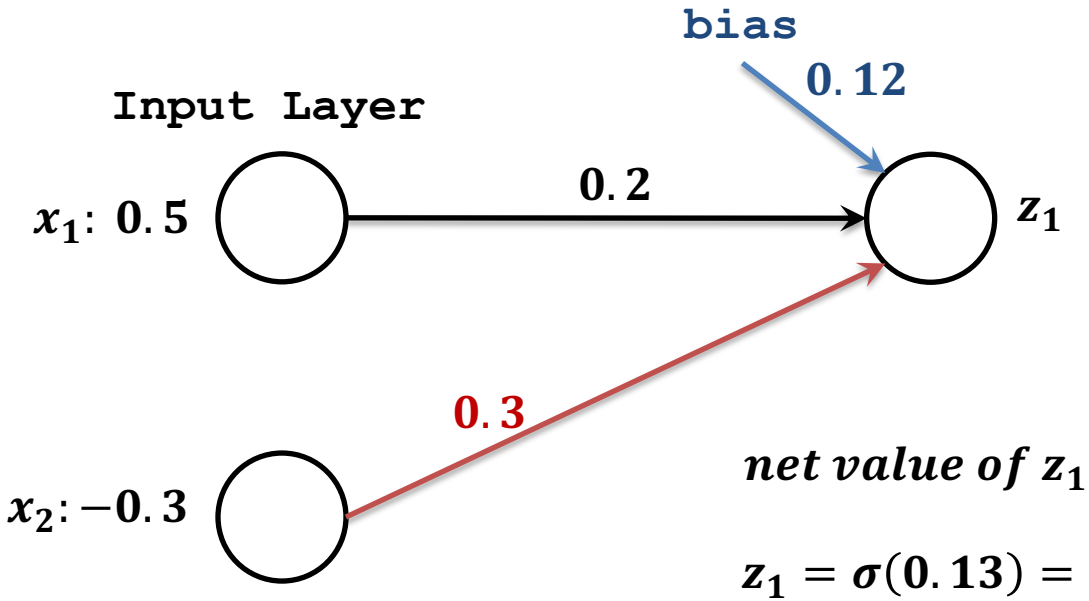


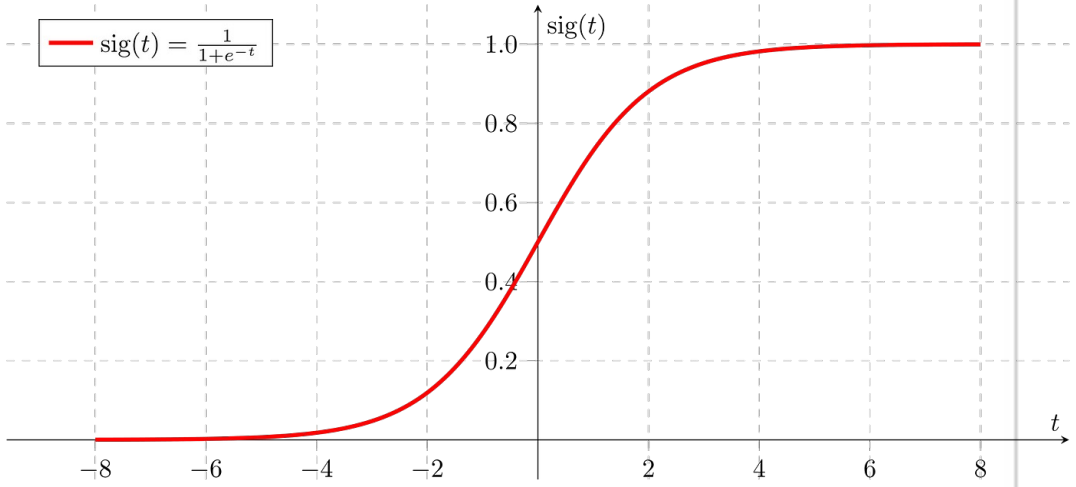
Activation function fires the net value, e.g. scaling it to range $[0, 1]$.



https://towardsdatascience.com/derivative-of-the-sigmoid-function-536880cf918e

Sigmoid: A Famous Activation Function

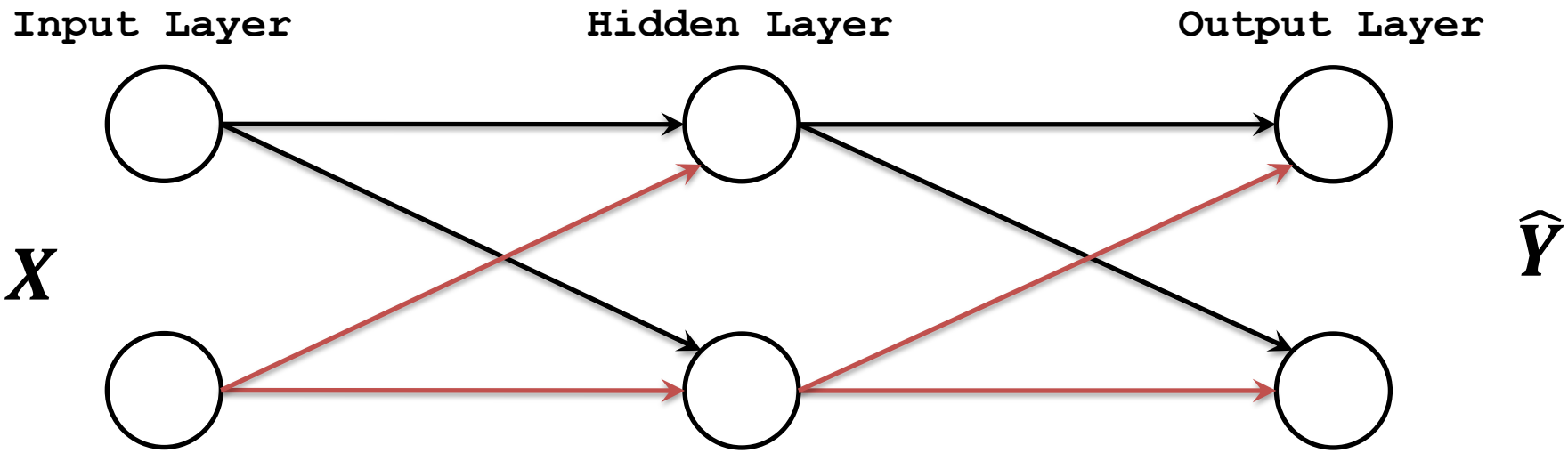




$$net\ value\ of\ z_1 = 0.13$$

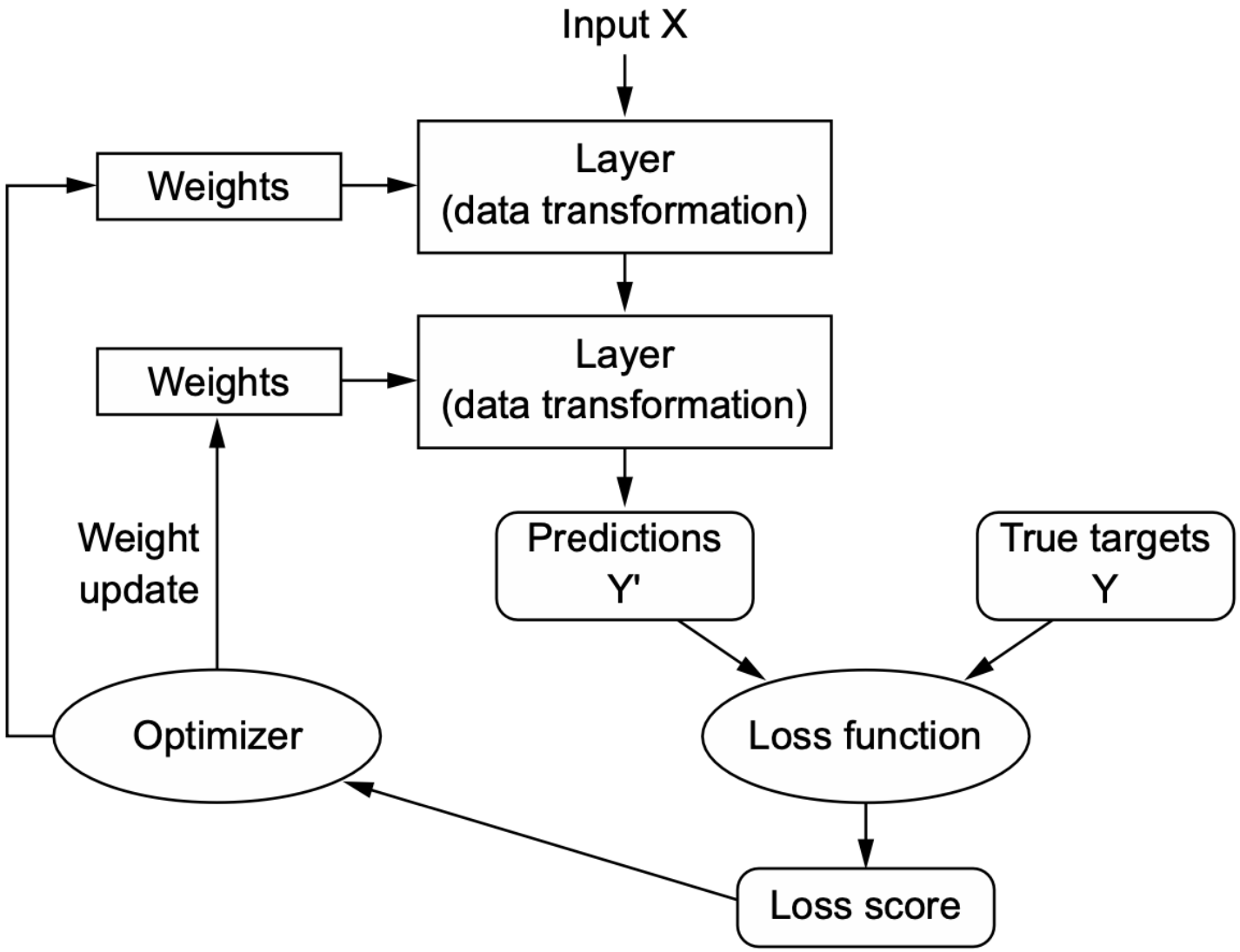
$$z_1 = \sigma(0.13) = 0.53$$

Forward Propagation



Error ----> Loss Function

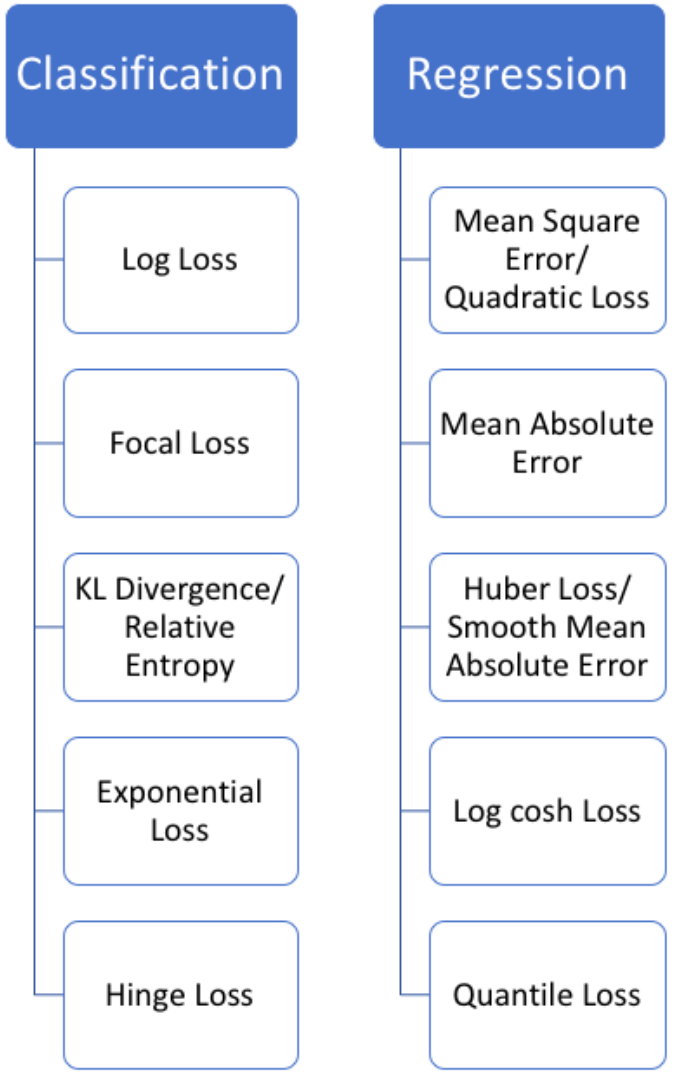
A Neural Network



Neural Network ingredient

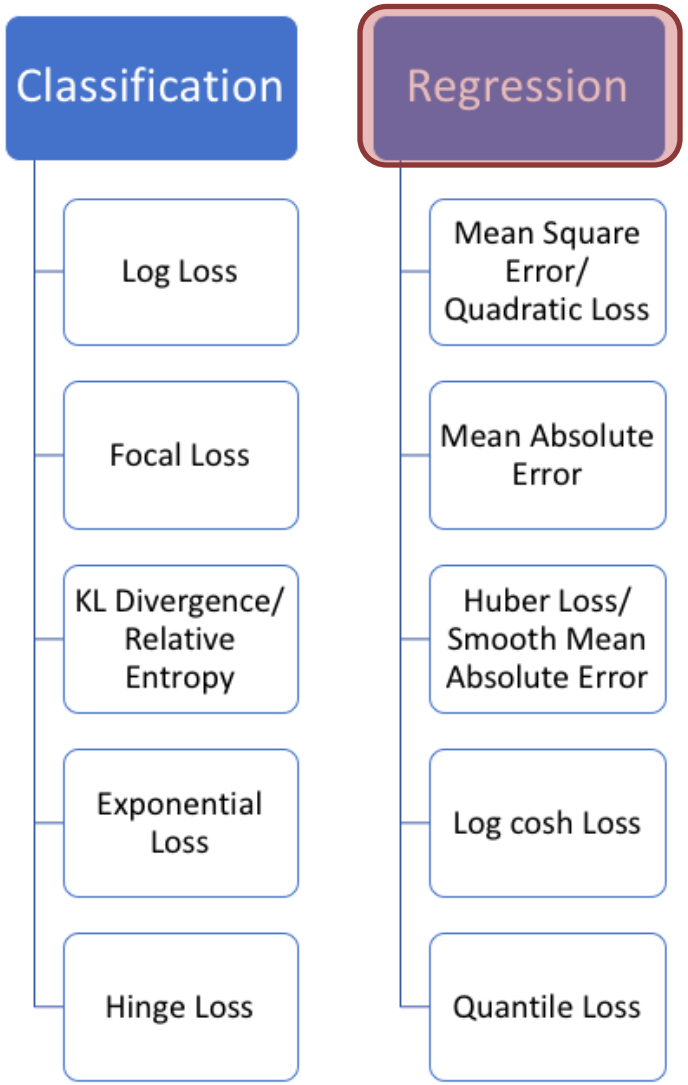
- Inputs
- Weights
- Some layers
- Some neurons in each layer
- Activation function for each neuron (better: each layer)
- Loss function
- Algorithms for minimizing loss function by updating weights (Optimizer)

Loss Functions



https://phuctrt.medium.com/loss-functions-why-what-where-or-when-189815343d3f

Loss Functions



https://phuctrt.medium.com/loss-functions-why-what-where-or-when-189815343d3f

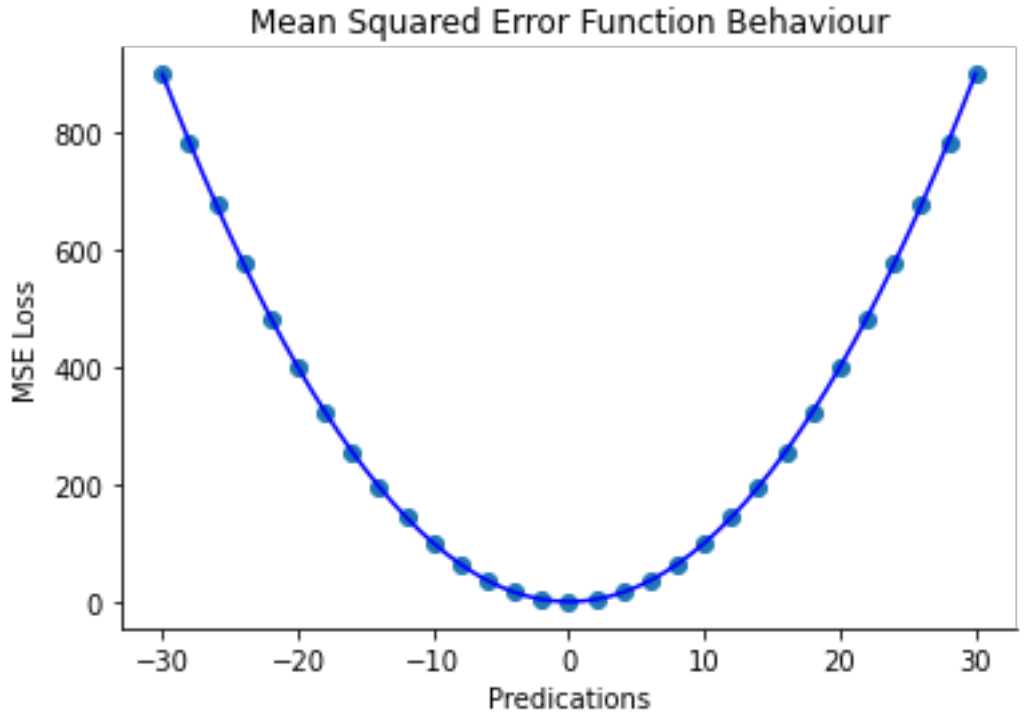
Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

n: *the number of samples*

Y_i: *target (true) data*

$\hat{Y}_i$: *predicted data*



https://datamonje.com/regression-loss-functions/

Mean Squared Error (MSE)

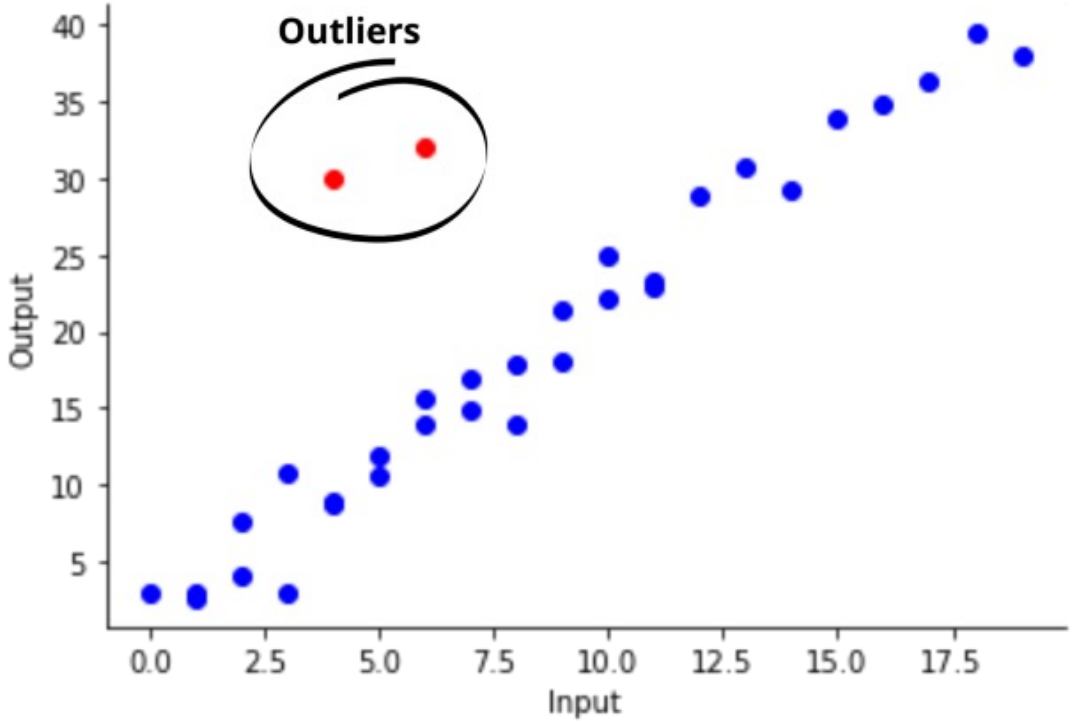
$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

n: *the number of samples*

Y_i: *target (true) data*

$\hat{Y}_i$: *predicted data*

- Very sensitive to outliers



https://datamonje.com/regression-loss-functions/

Mean Squared Error (MSE)

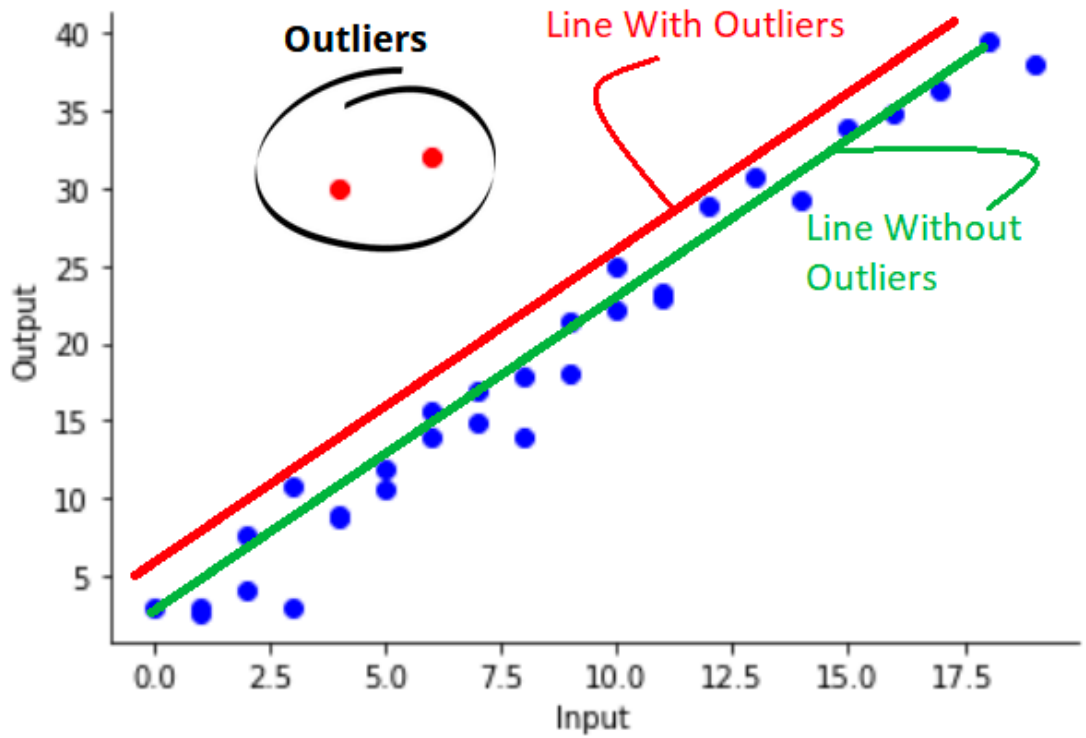
$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

n: *the number of samples*

Y_i: *target (true) data*

$\hat{Y}_i$: *predicted data*

- Very sensitive to outliers



https://datamonje.com/regression-loss-functions/

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2}$$

n: *the number of samples*

Y_i: *target (true) data*

$\widehat{Y}_i$: *predicted data*

Root Mean Squared Error (RMSE)

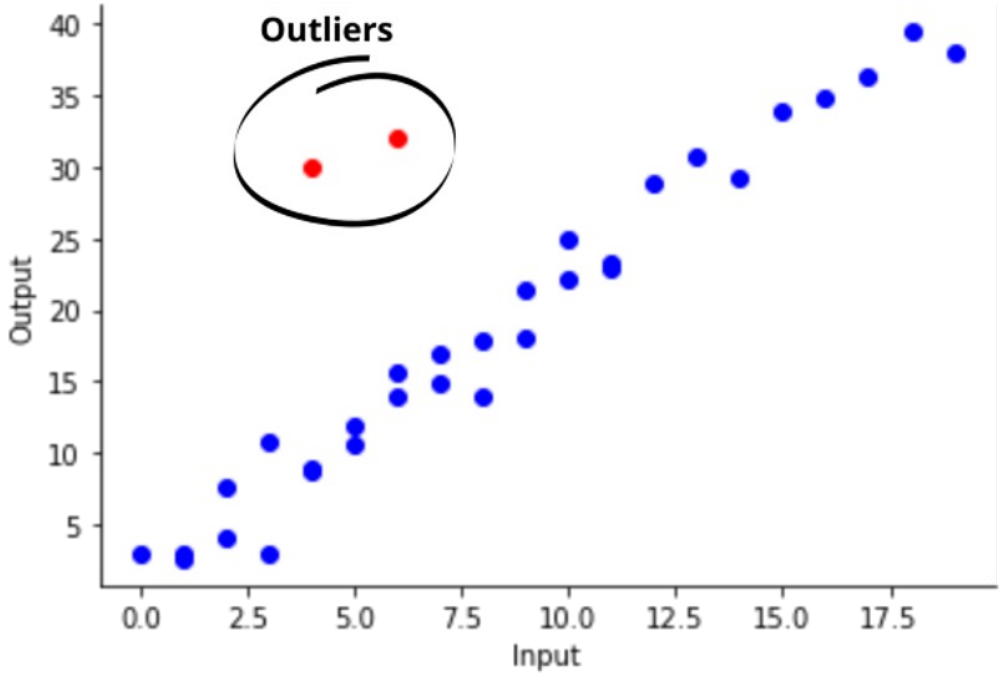
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2}$$

n: *the number of samples*

Y_i: *target (true) data*

$\hat{Y}_i$: *predicted data*

- Very sensitive to outliers



https://datamonje.com/regression-loss-functions/

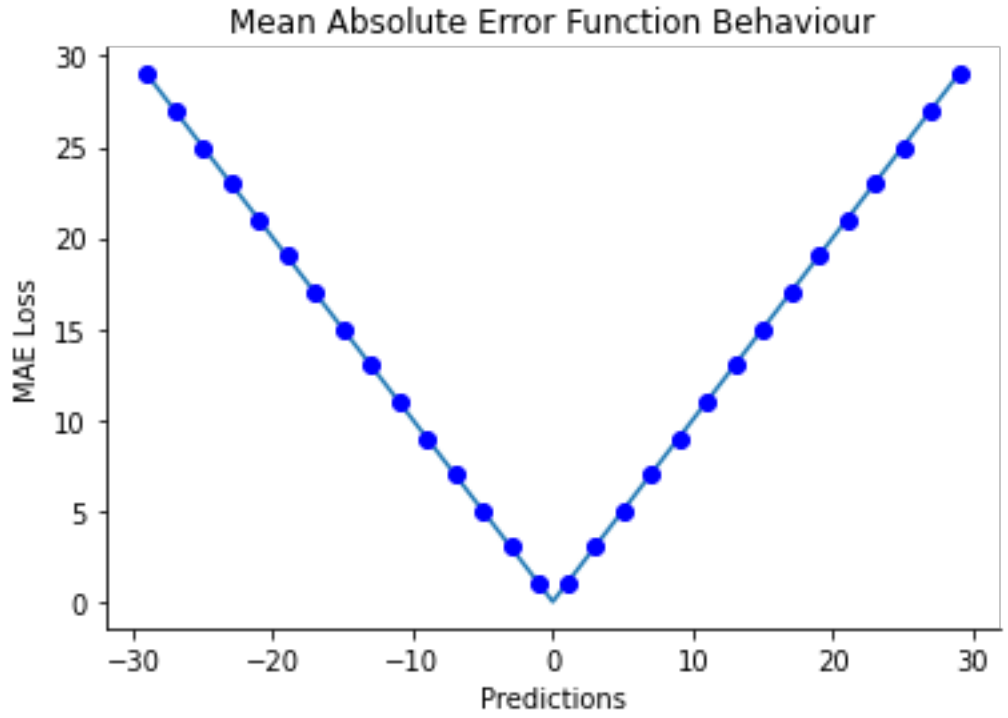
Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \widehat{Y}_i\right|$$

n: *the number of samples*

Y_i: *target (true) data*

$\widehat{Y}_i$: *predicted data*



https://datamonje.com/regression-loss-functions/

Mean Absolute Error (MAE)

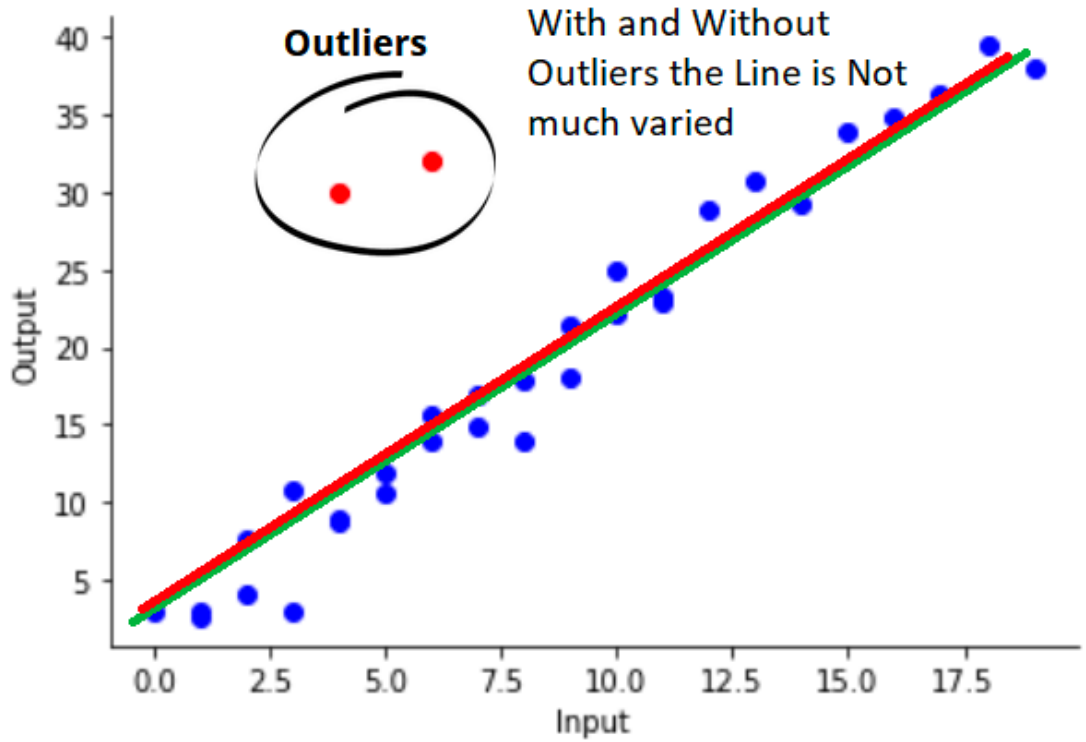
$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right|$$

n: *the number of samples*

Y_i: *target (true) data*

$\hat{Y}_i$: *predicted data*

- The Optimization is a little bit complex compared to MSE

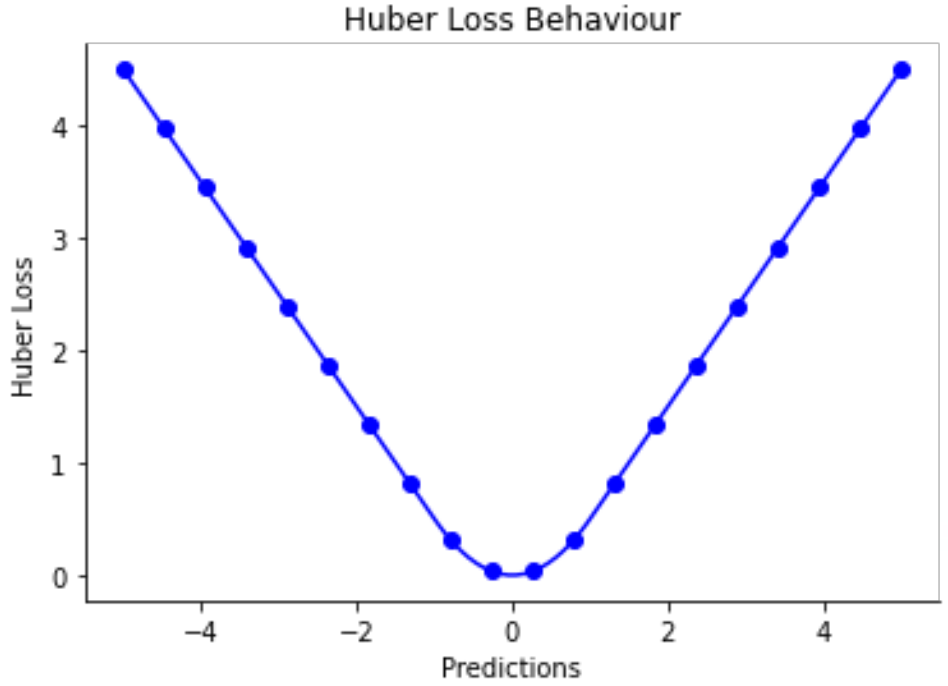


https://datamonje.com/regression-loss-functions/

Huber Loss

$$L_\delta(Y_i, \hat{Y}_i) = \begin{cases} \frac{1}{2}(Y_i - \hat{Y}_i)^2, & |Y_i - \hat{Y}_i| \le \delta \\ \delta\left(|Y_i - \hat{Y}_i| - \frac{1}{2}\delta\right), & |Y_i - \hat{Y}_i| > \delta \end{cases}$$

$$HuberLoss = \frac{1}{n}\sum_{i=1}^{n} L_\delta(Y_i, \hat{Y}_i)$$



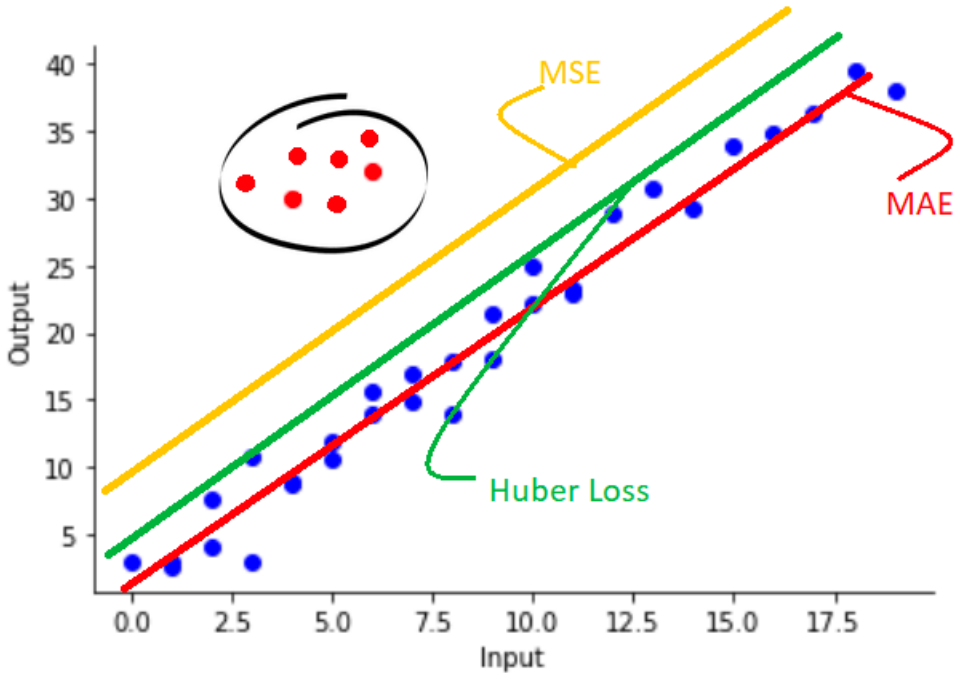
https://datamonje.com/regression-loss-functions/

Huber Loss

$$L_\delta(Y_i, \hat{Y}_i) = \begin{cases} \frac{1}{2}(Y_i - \hat{Y}_i)^2, & |Y_i - \hat{Y}_i| \le \delta \\ \delta\left(|Y_i - \hat{Y}_i| - \frac{1}{2}\delta\right), & |Y_i - \hat{Y}_i| > \delta \end{cases}$$

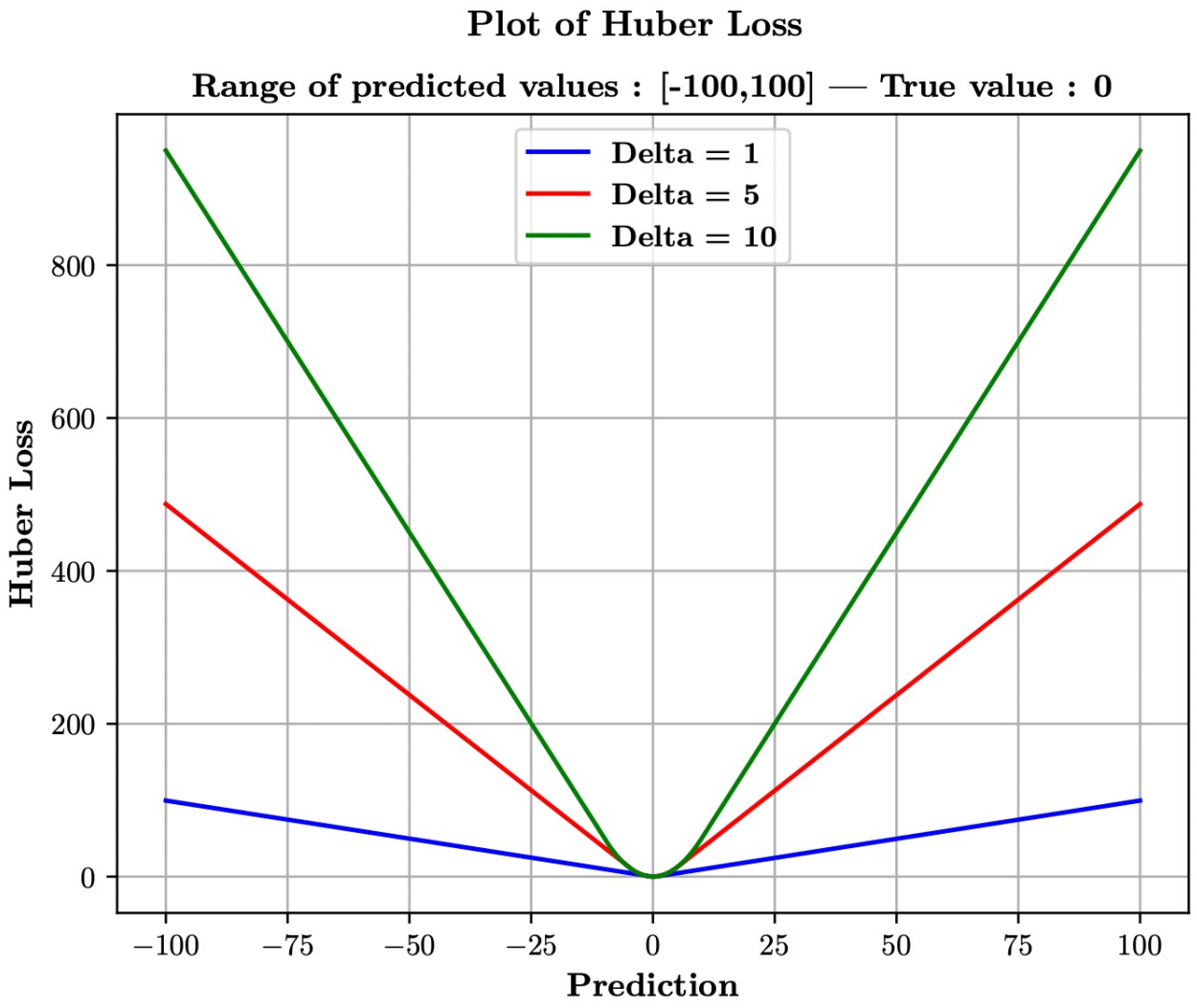
$$HuberLoss = \frac{1}{n}\sum_{i=1}^{n} L_\delta(Y_i, \hat{Y}_i)$$

- The equation is a bit complex
- we also need to adjust the δ based on our requirement



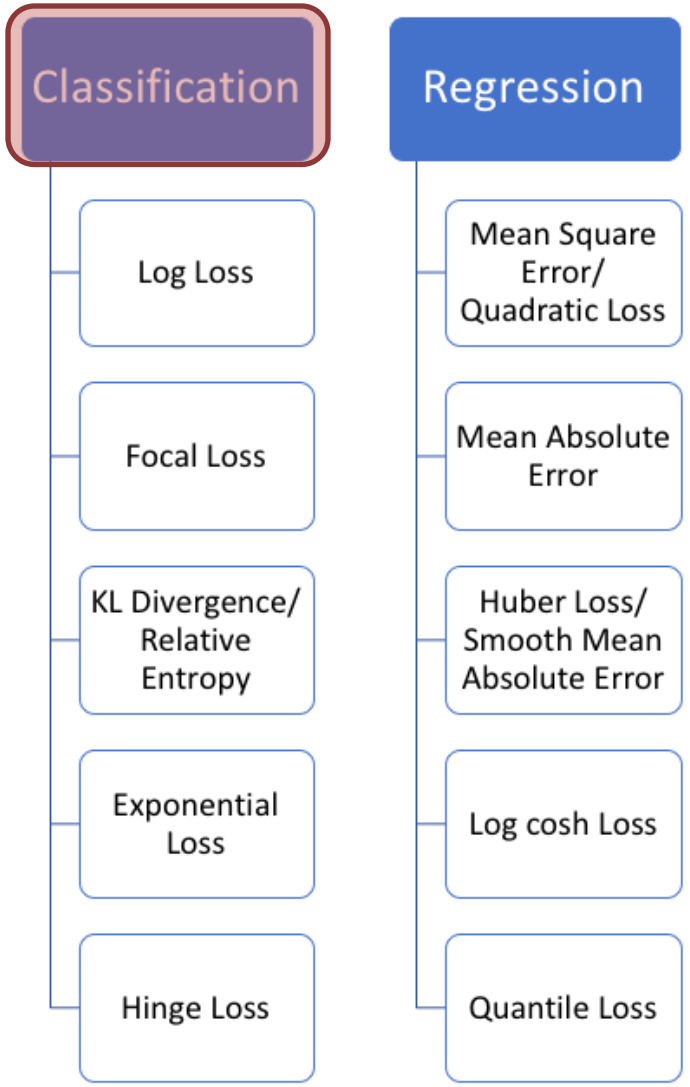
https://datamonje.com/regression-loss-functions/

Huber Loss



https://datasciencepreparation.com/blog/articles/what-loss-functions-can-be-used-for-regression-which-one-is-better-for-outliers/

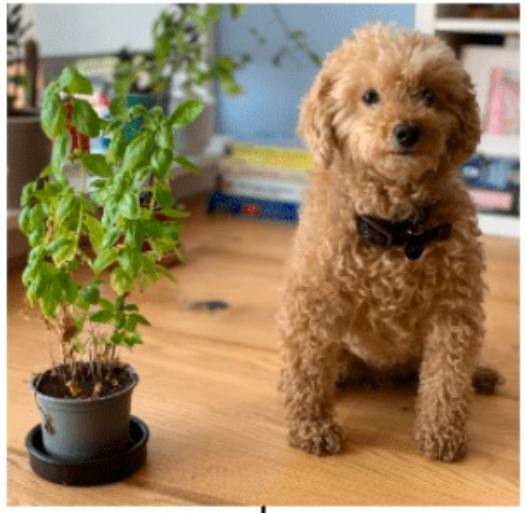
Loss Functions



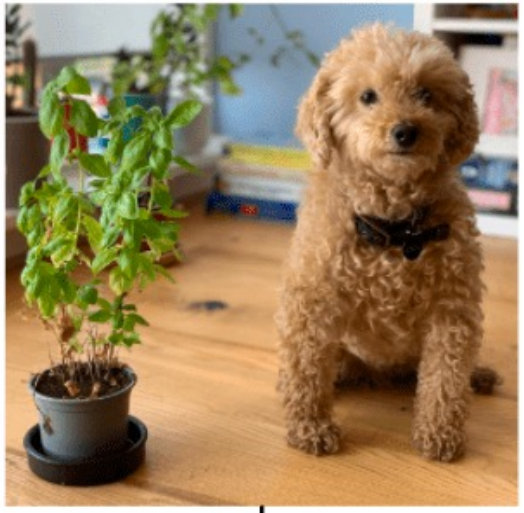
https://phuctrt.medium.com/loss-functions-why-what-where-or-when-189815343d3f

3 Types of Classification

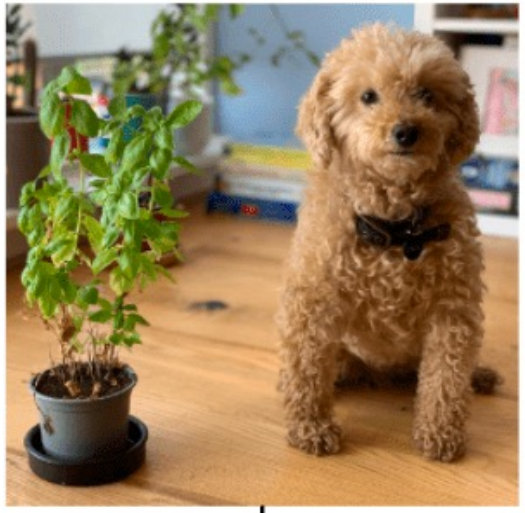
Binary Classification



Multiclass Classification



Multilabel Classification



https://www.mathworks.com/help/deeplearning/ug/multilabel-image-classification-using-deep-learning.html

Binary Cross Entropy

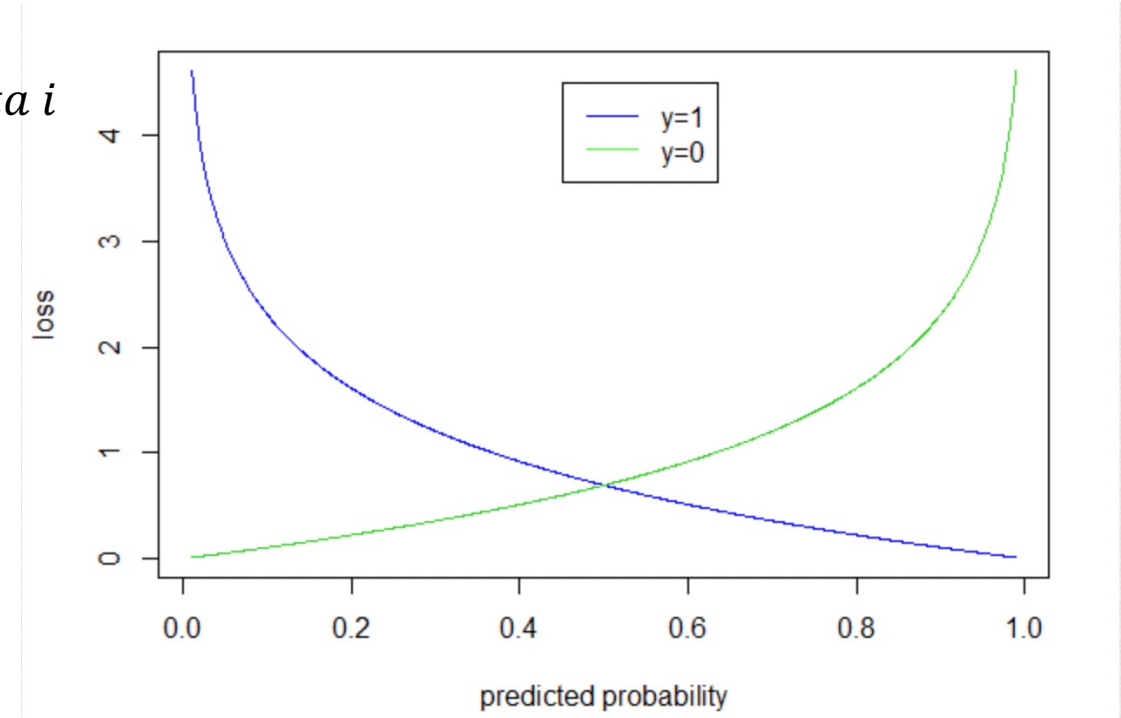
- Widely used for 2 classes

$$CrossEntropy = -\frac{1}{n}\left[\sum_{i=1}^{n} Y_i \log p_i + (1 - Y_i)\log(1 - p_i)\right]$$

n: *the number of samples*

Y_i: *target (true) class label* (0 *or* 1)

p_i: *predicted probability for the class of data i*



https://www.datasciencepreparation.com/blog/articles/what-is-cross-entropy-loss/

Cross Entropy for Multi-label Classification

$$CrossEntropy = -\frac{1}{n}\sum_{j=1}^{n}\left[\sum_{i=1}^{c} Y_i \log p_i + (1 - Y_i)\log(1 - p_i)\right]$$

c: *the number of classes*

n: *the number of samples*

Y_i: *target (true) class label* (0 *or* 1)

p_i: *predicted probability for the class of data i*


THANK
YOU !