DEEP LEARNING

BY M.J.PASSLAR

COMPUTERONIC – TEHRAN – IRAN

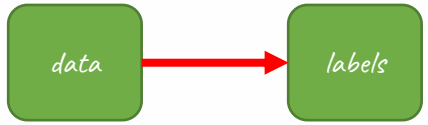
2022

CHAPTER 2 : OVER VIEW

Learning methods ...

Supervised learning

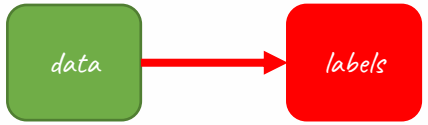
Most common learning method



- Sequence generator
- Syntax-tree generator
- Object detection
- Image segmentation

Unsupervised learning

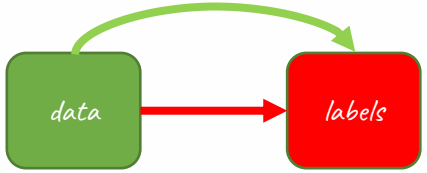
Most common in problem with limit known labels



- Dimension reduction
- clustering

Self-supervised learning

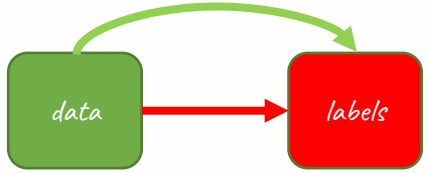
Most common in auto encoders



- Auto encoders

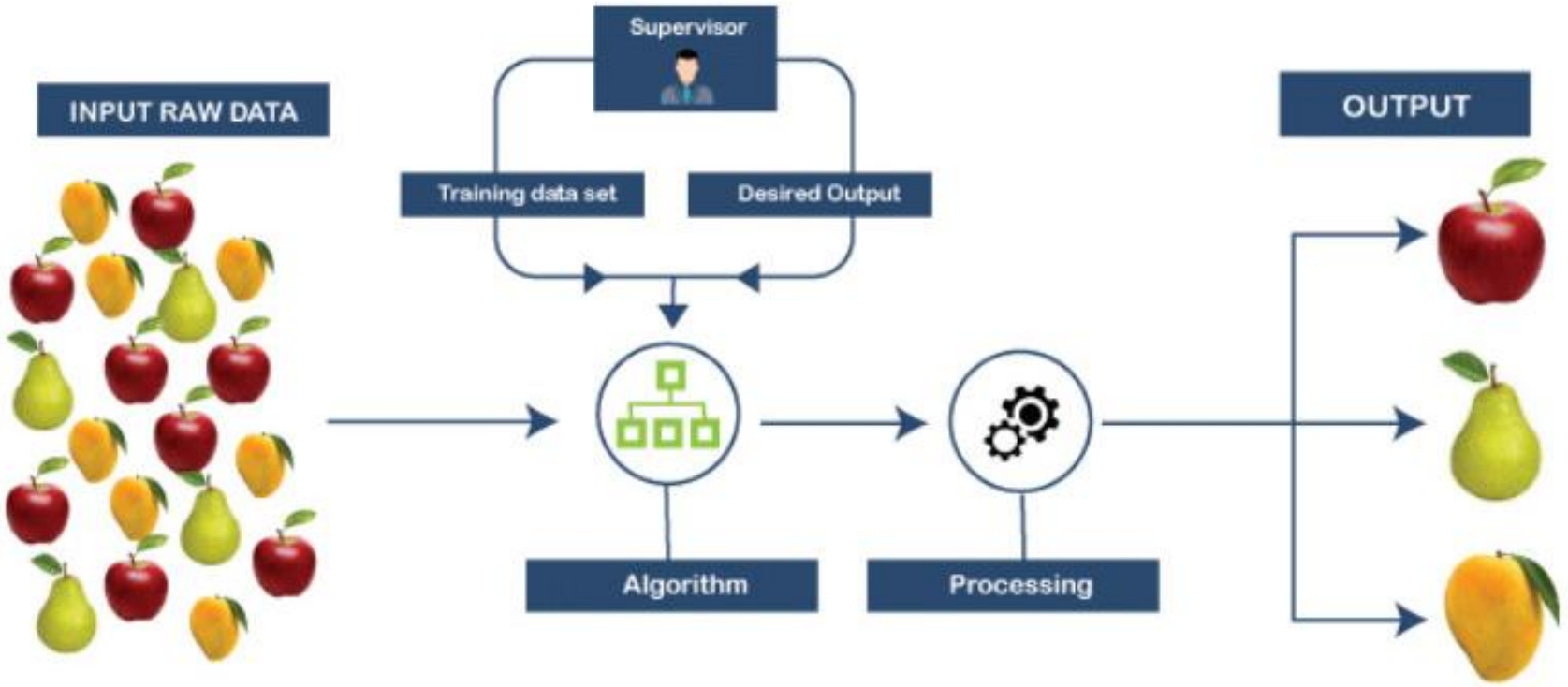
Reinforcement learning

Most common in Reinforcement clustering

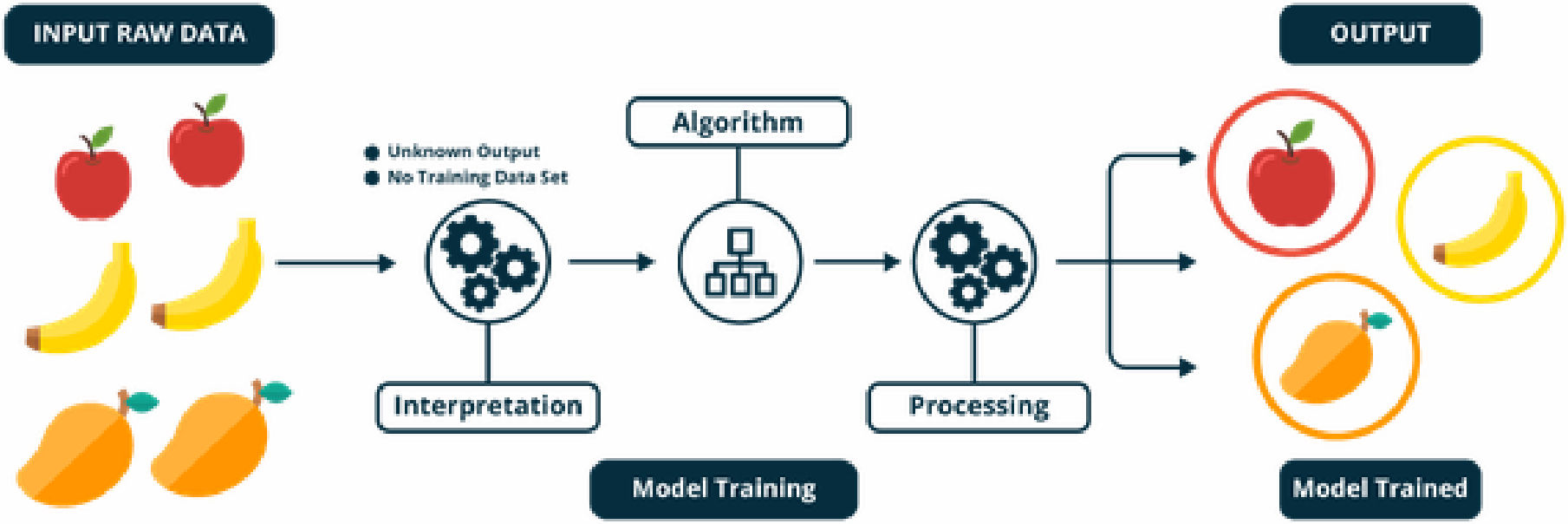


- Reinforcement clustering

Supervised learning method

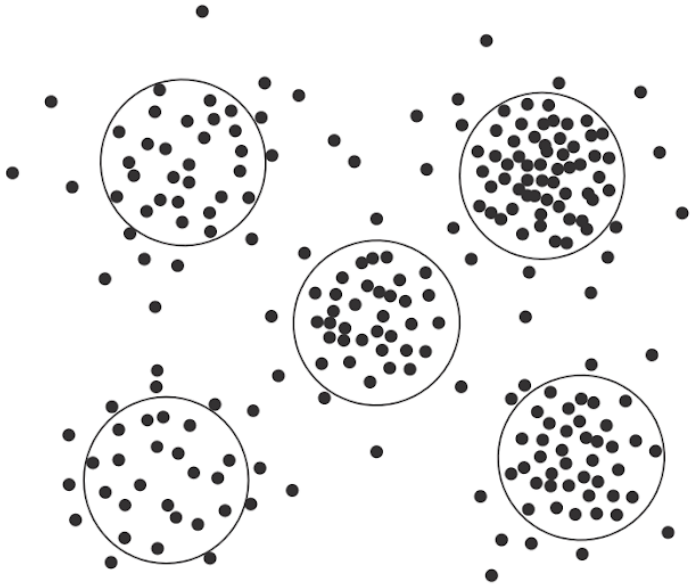


Unsupervised learning method

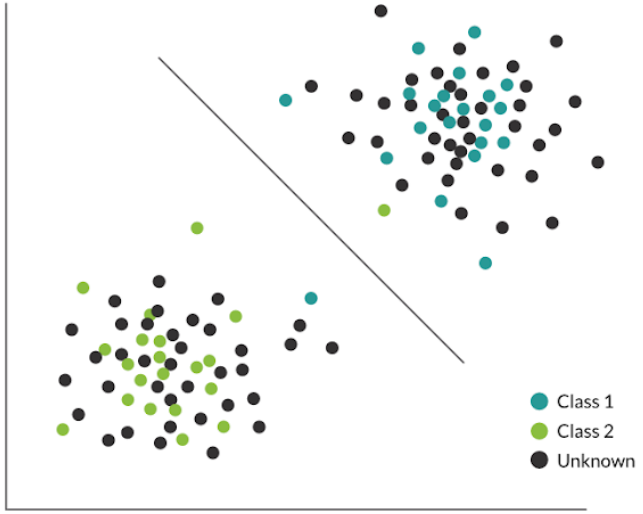


Supervised vs unsupervised

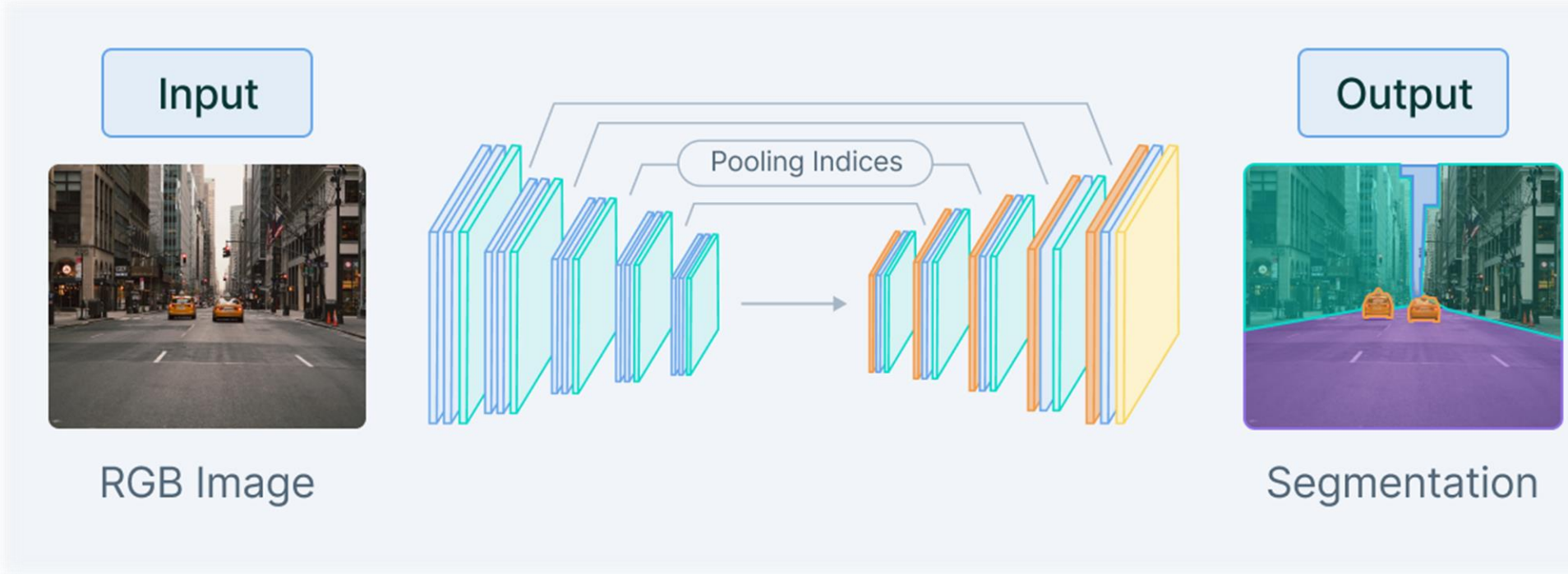
Unsupervised



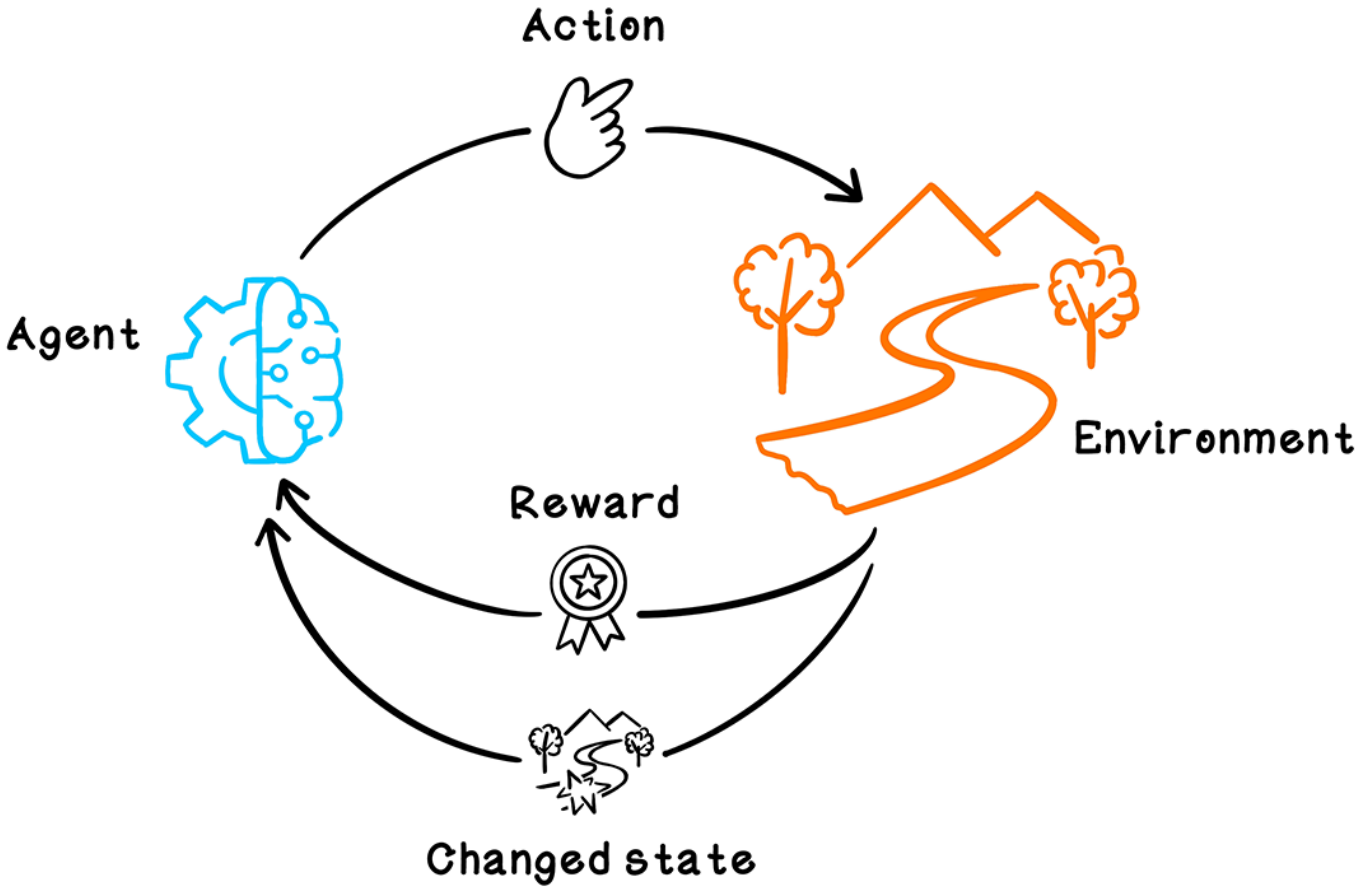
Supervised



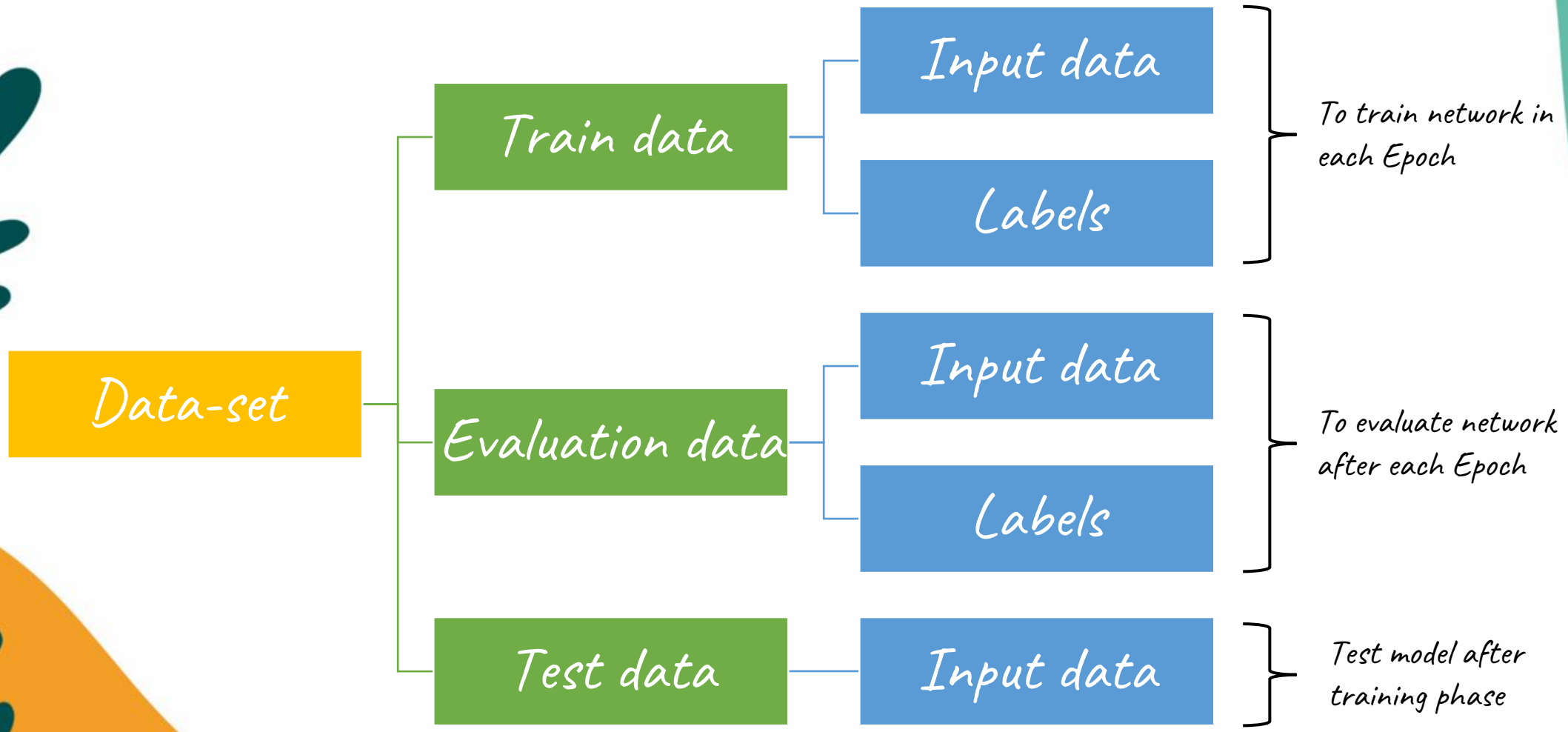
Auto encoder learning method(Self- Supervised method)



reinforcement learning method



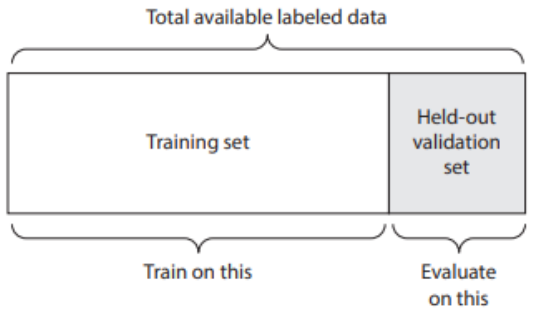
Test and Evaluation methods



Evaluation Methods ...

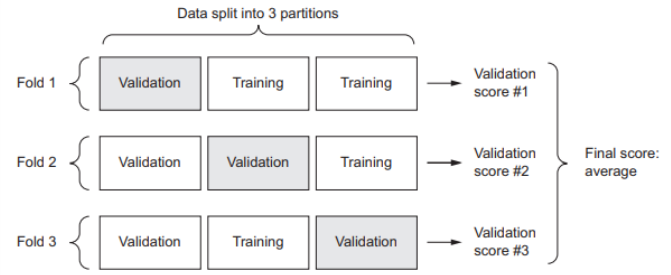
Simple hold-out validation

- Simplest method
- Useful for large data sets
- Trained model is not valid
- May overfitting happened



K-fold validation

- Use for small data sets
- Trained model is valid (most the time)



K-fold validation with shuffling

- Most valid method
- Use in final NN validation

- Use K-fold method but with shuffled data

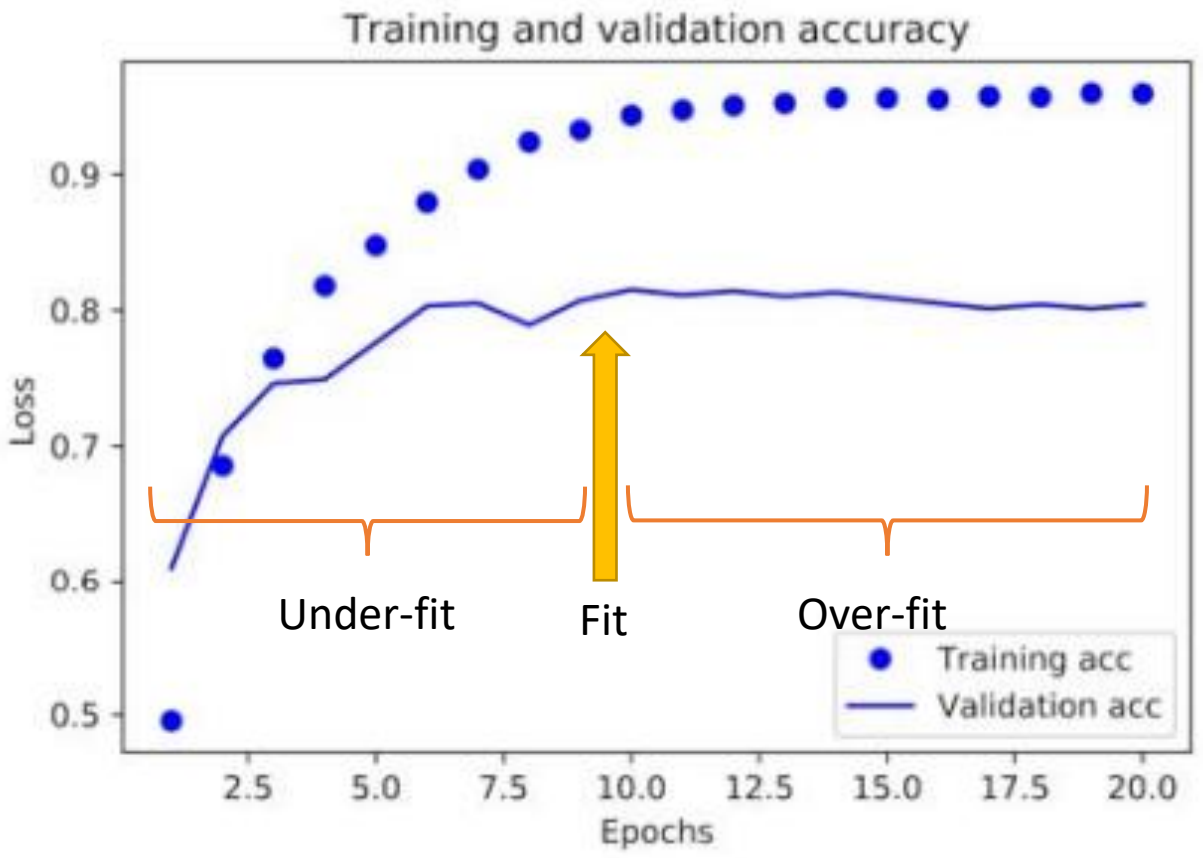
Data post-processing ...

Vectorization	• Convert all data and labels to Tensor • Choice validation method and spilt data
Normalization	• Normalize all data to • Make all data homogenous (all data should have same range)
Handling missing data	• Replace missing data with ZERO • Train and test data both should have some missing data
Feature engineering	• Making problem easier • -> less recourses + less data needed

Feature engineering

Raw data: pixel grid			Large and complex network needed
Better features: clock hands' coordinates	{x1: 0.7, y1: 0.7} {x2: 0.5, y2: 0.0}	{x1: 0.0, y2: 1.0} {x2: -0.38, 2: 0.32}	Large and simple network needed
Even better features: angles of clock hands	theta1: 45 theta2: 0	theta1: 90 theta2: 140	small and simple network needed

Under-fit, fir & Over-fit



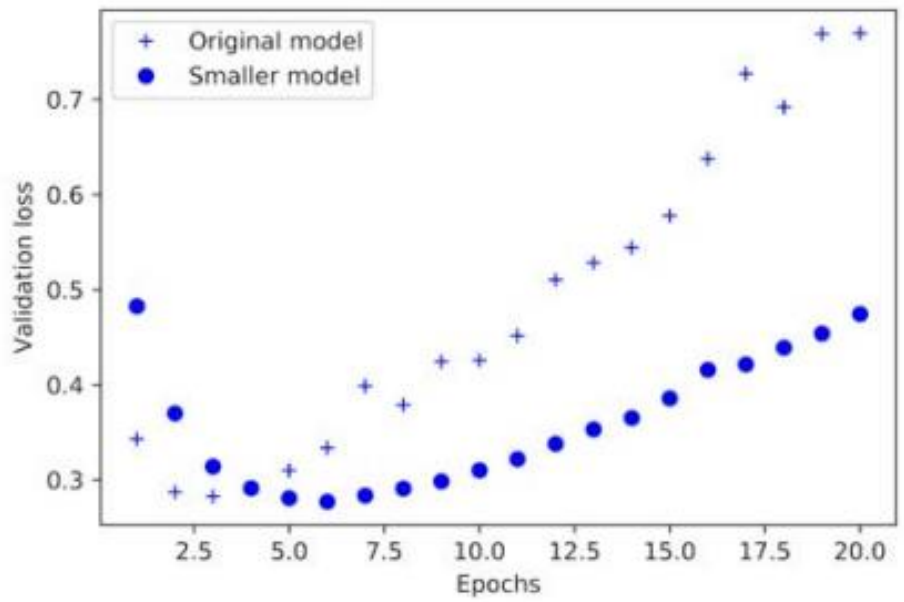
How to prevent over-fitting ?

The best solution is to train network with bigger dataset

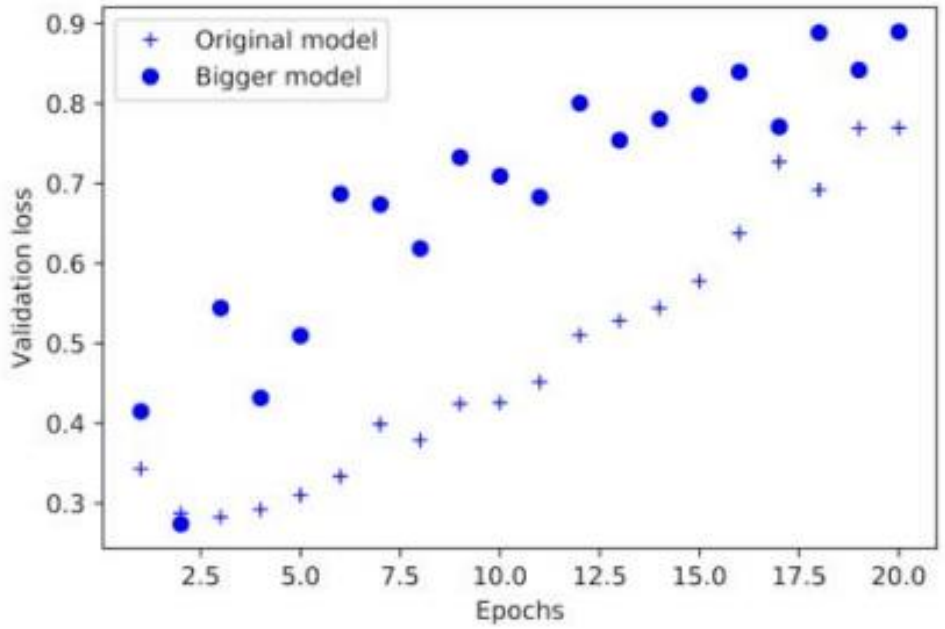
Indirect method to make more data :

- Reduce network size
  - Change system memories capacity
- Adding weight regularization factor
  - L1 method
  - L2 method
- Use dropout method
  - Make some weighted zero

Changing network size

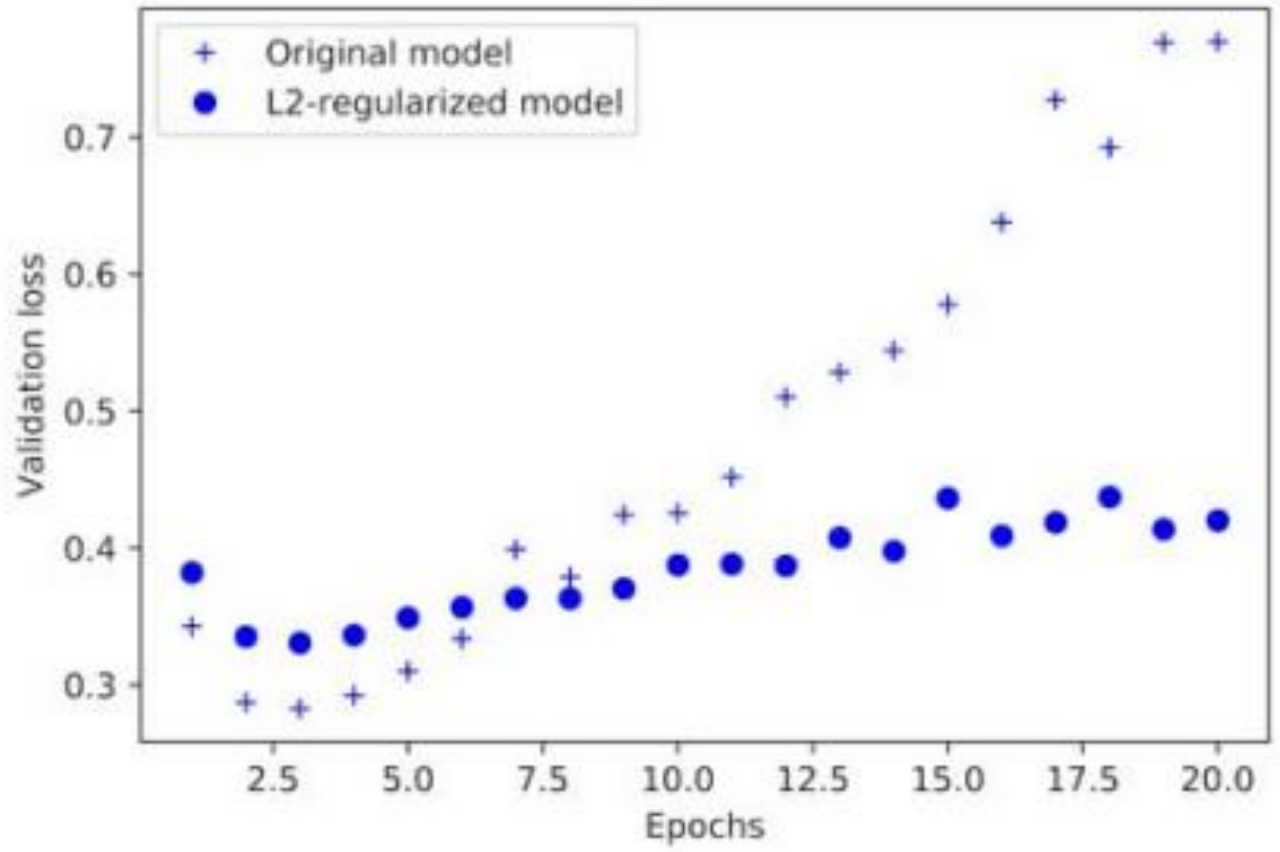


Smaller network



Bigger network

Add L1 & L2 regularization to network



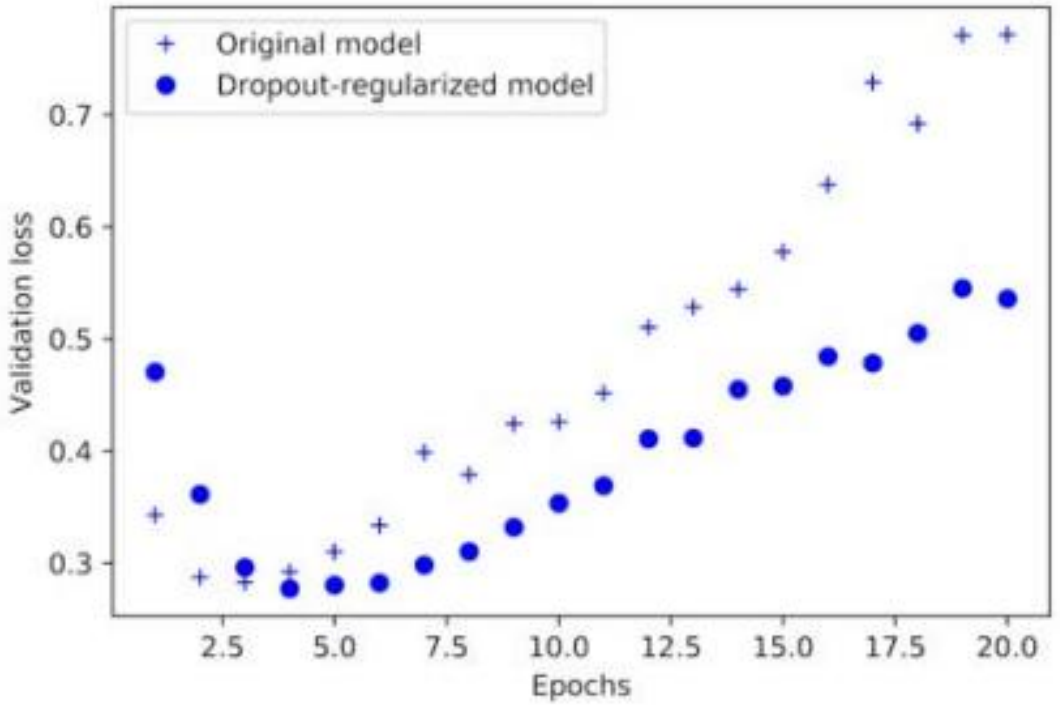
Add dropout to network

0.3	0.2	1.5	0.0
0.6	0.1	0.0	0.3
0.2	1.9	0.3	1.2
0.7	0.5	1.0	0.0

50% dropout

0.0	0.2	1.5	0.0
0.6	0.1	0.0	0.3
0.0	1.9	0.3	0.0
0.7	0.0	0.0	0.0

* 2



How networks calculate weight ? (activation functions)

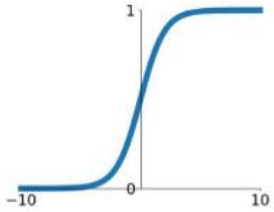
Problem type	Last-layer activation	Loss function
Binary classification	`sigmoid`	`binary_crossentropy`
Multiclass, single-label classification	`softmax`	`categorical_crossentropy`
Multiclass, multilabel classification	`sigmoid`	`binary_crossentropy`
Regression to arbitrary values	None	`mse`
Regression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

Activation function

a mathematical function that converts a vector of numbers into a vector of probabilities

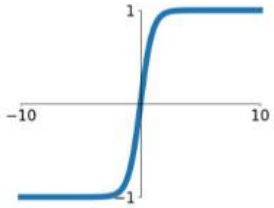
Sigmoid

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



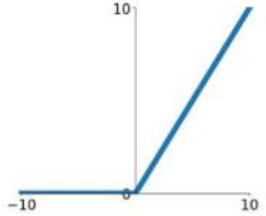
tanh

$$\tanh(x)$$



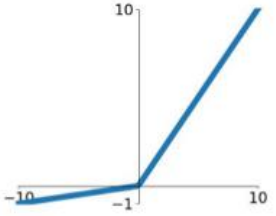
ReLU

$$\max(0, x)$$



Leaky ReLU

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

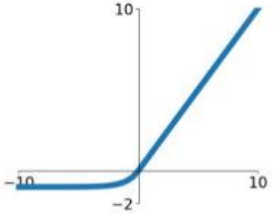


Maxout

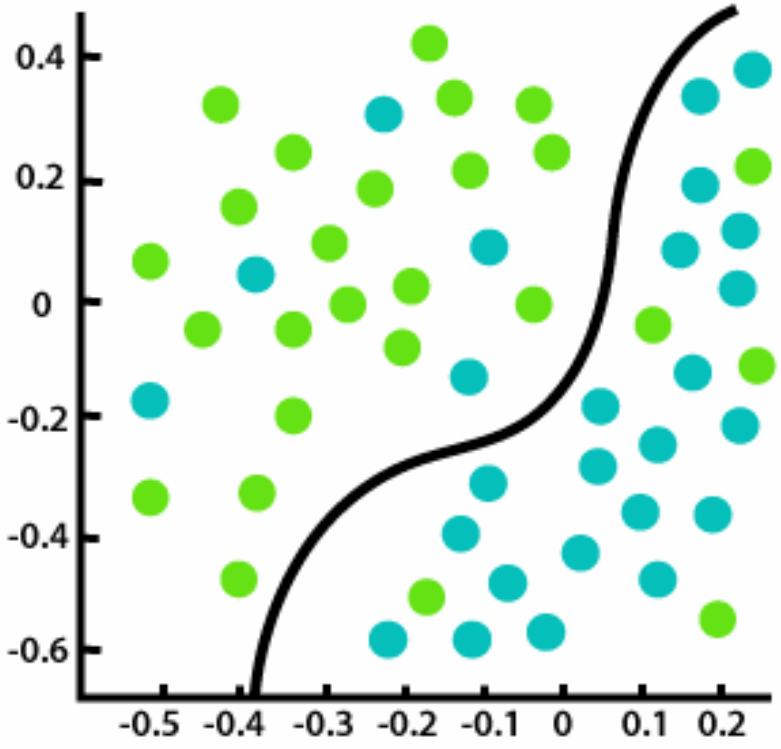
$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

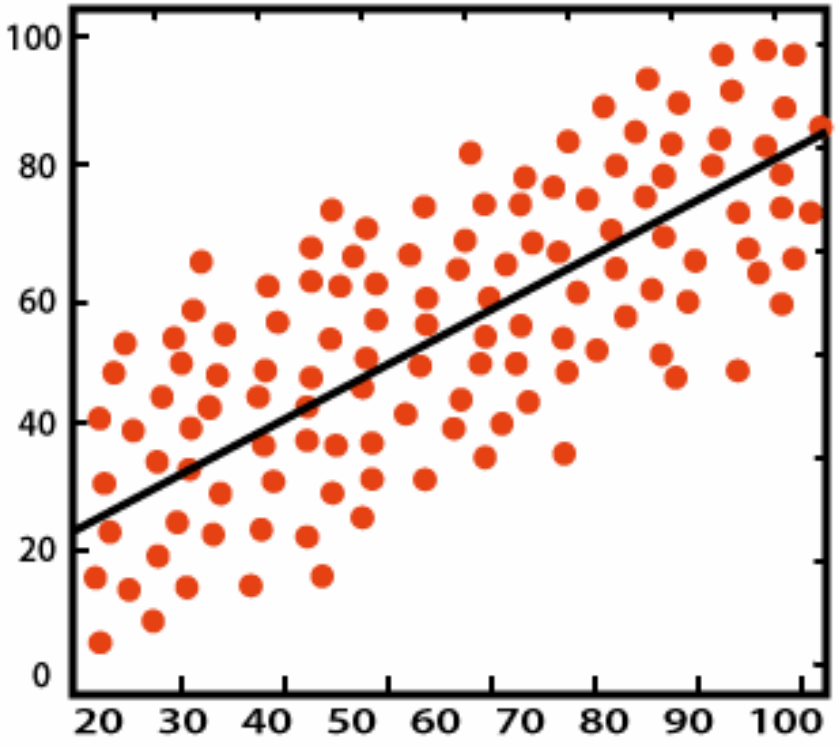
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Classification Vs Regression problems

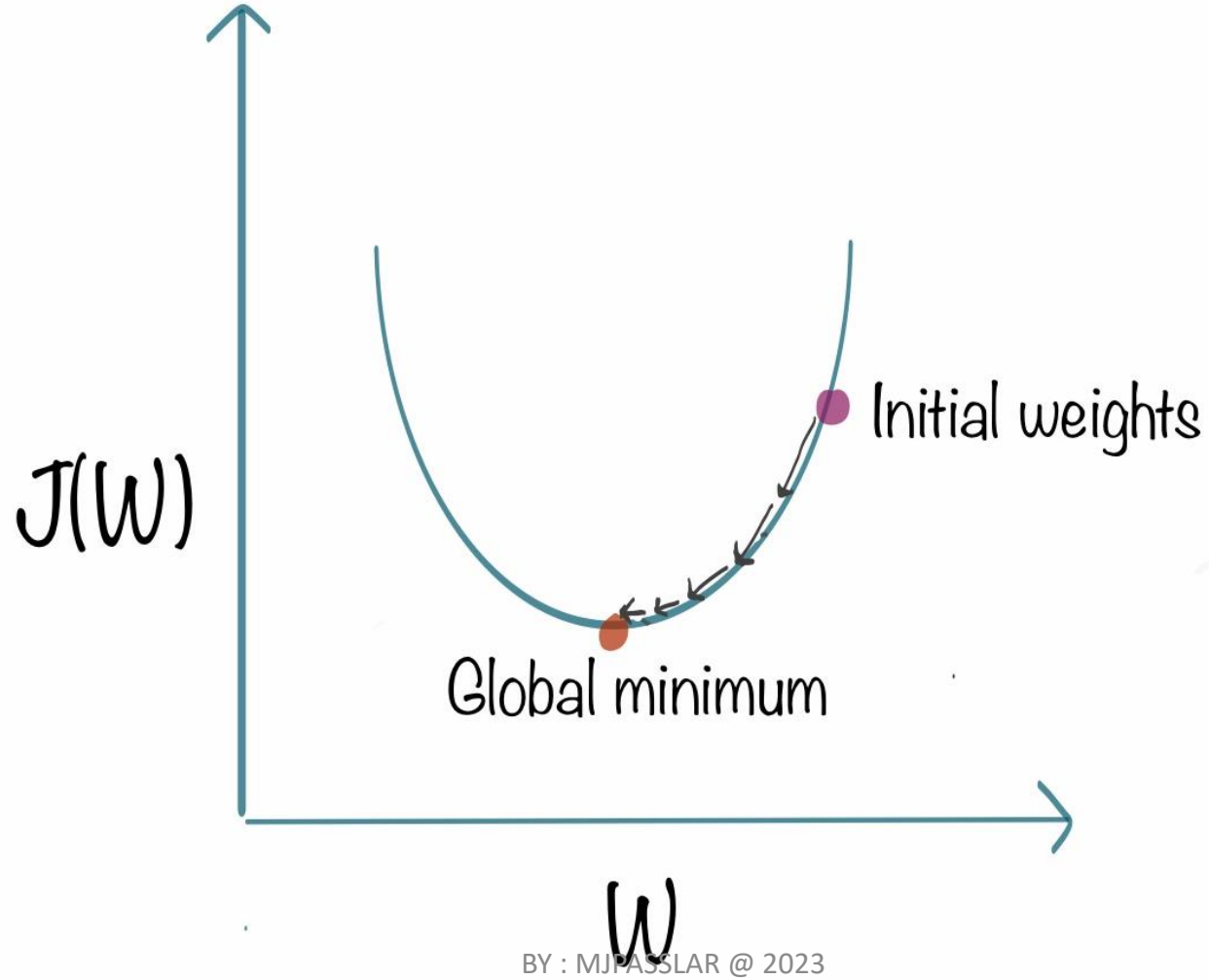


Classification



Regression

Mse,Binary crossentropy and categorical crossentropy



Deep learning problem work-flow