DEEP LEARNING WITH PYTORCH

Presented by

Bharathi Raja Asoka Chakravarthi

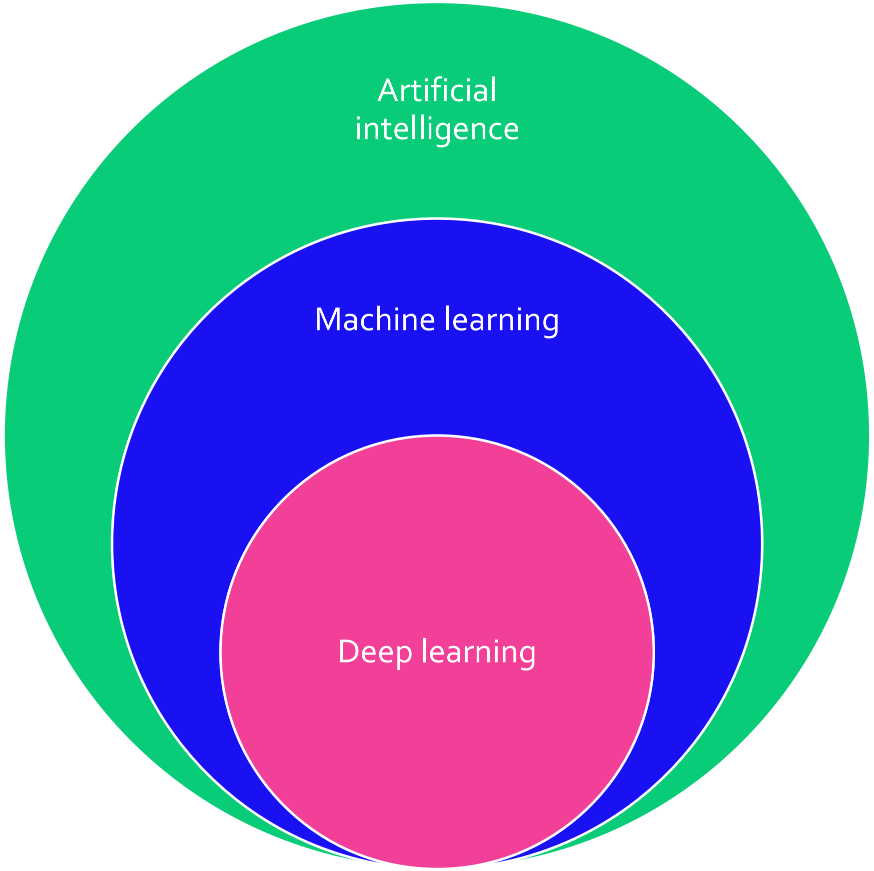
Shardul Suryawanshi

Bharath Sudharsan

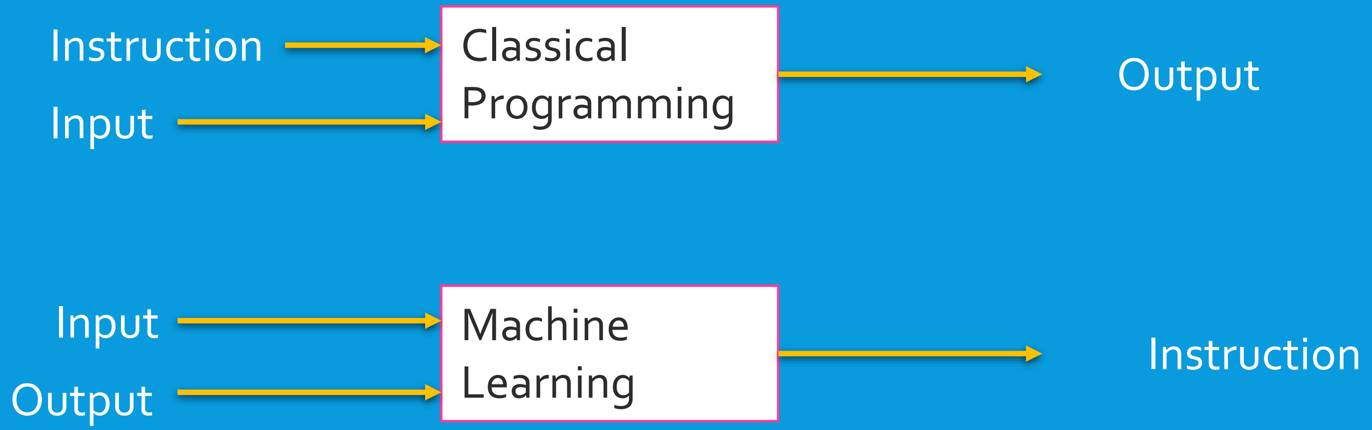
OUTLINE

- Introduction: What is deep learning?
- Building blocks of neural network
- Fundamentals of machine learning
  - Training, Validation, and Test;
  - Overfitting and Underfitting
- Practical session: Bharathi, Shardul, and Bharath

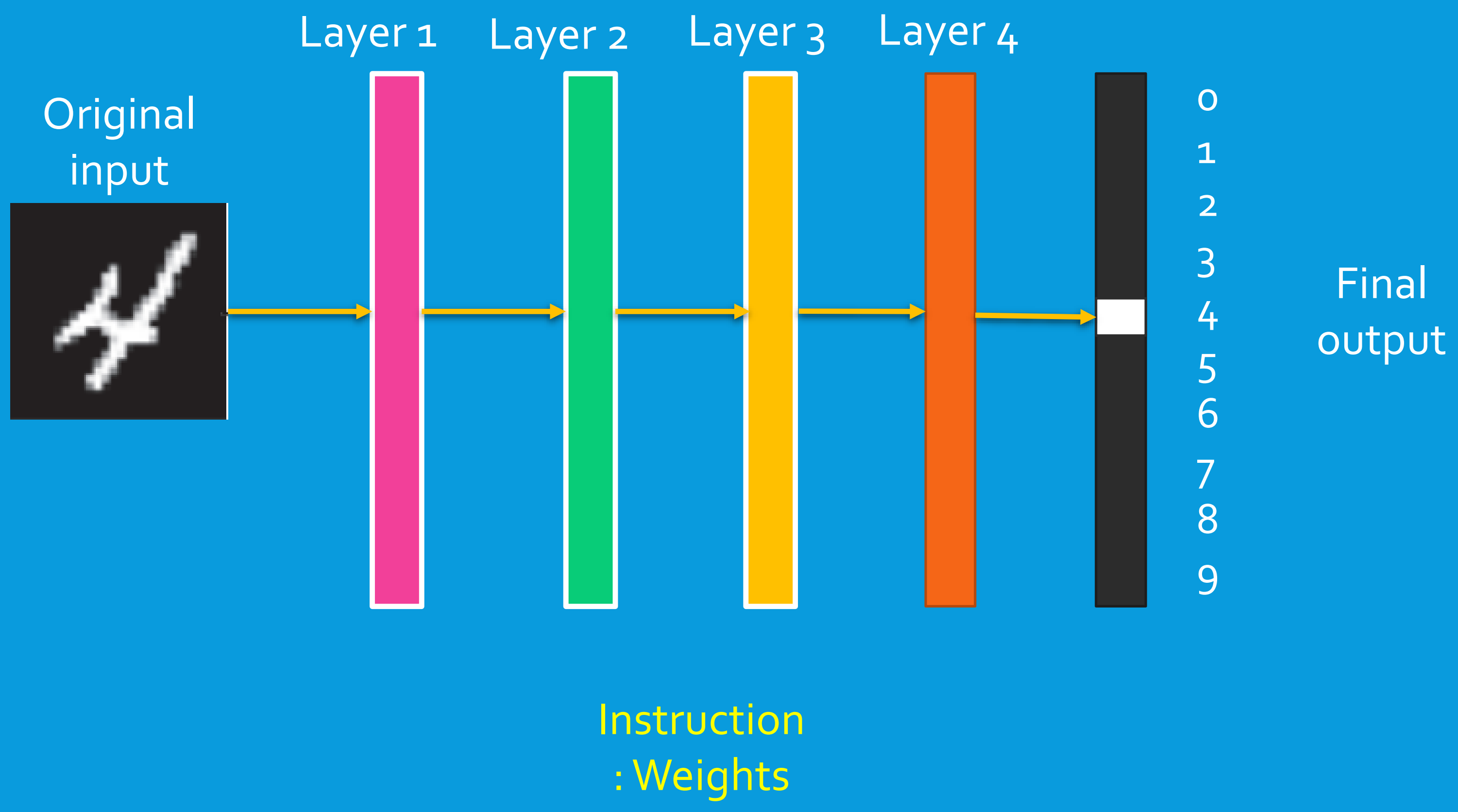
WHAT IS DEEP LEARNING?



MACHINE LEARNING

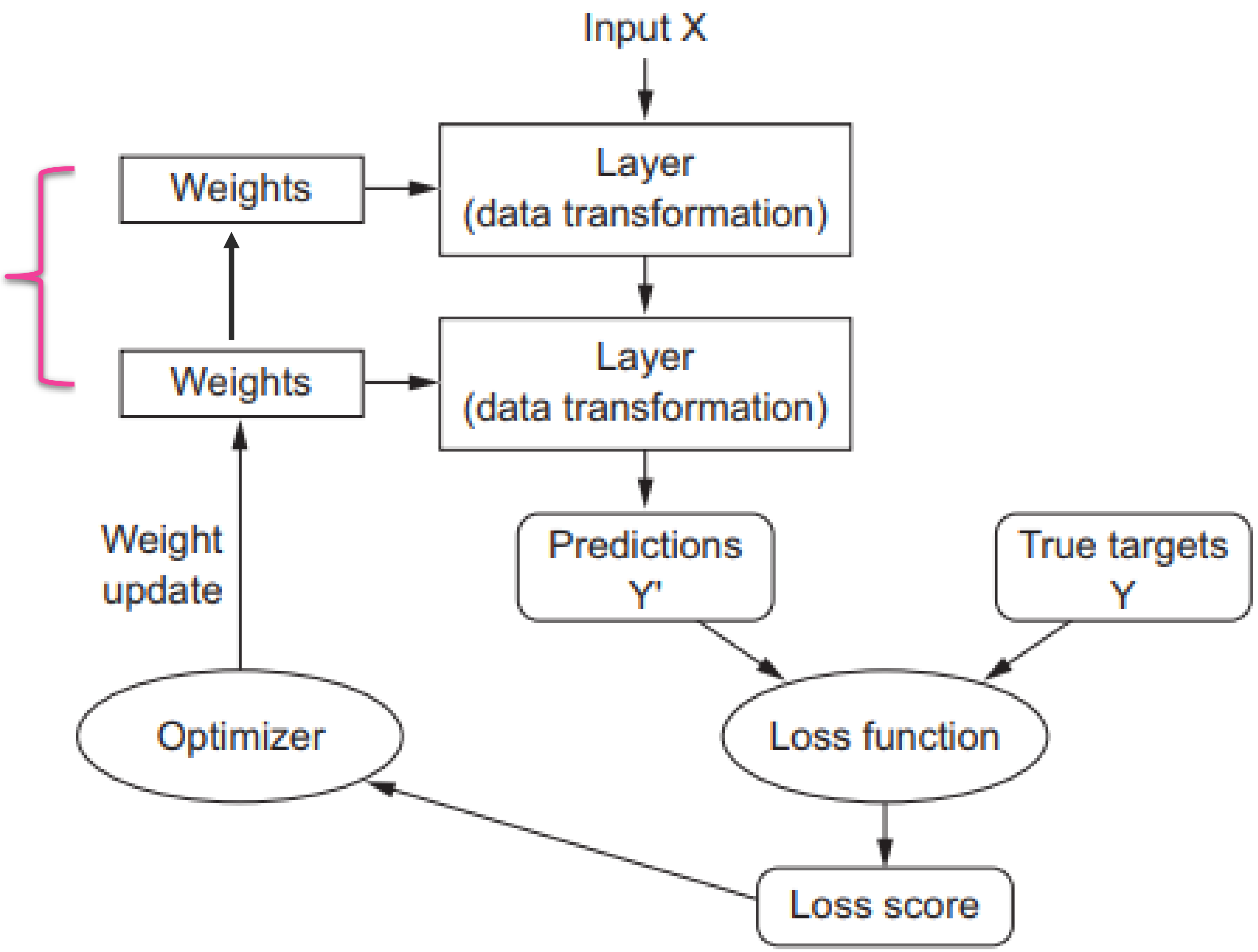


DEEP LEARNING



DEEP LEARNING

- Goal: Finding the Instruction -- right values for the weights

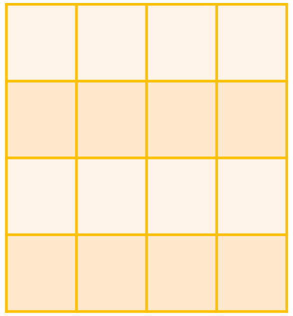


MATHEMATICAL BUILDING BLOCKS OF NEURAL NETWORK

- A tensor is a generalization of vectors and matrices to potentially higher dimensions.
- Scalar (0D tensors)

- Vectors (1D tensors)

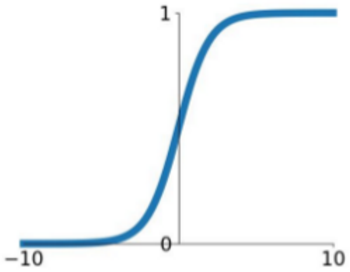
- Matrices (2D tensors)



Activation Functions

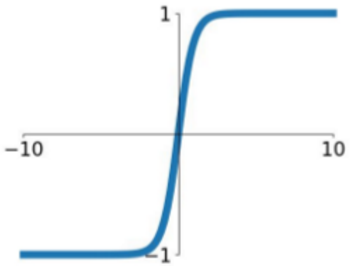
Sigmoid

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



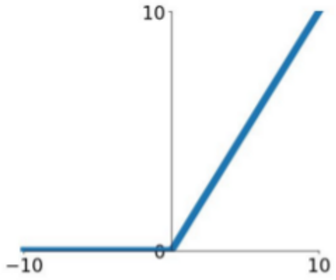
tanh

$\tanh(x)$



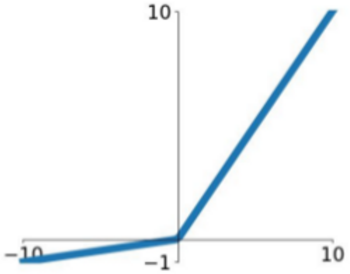
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

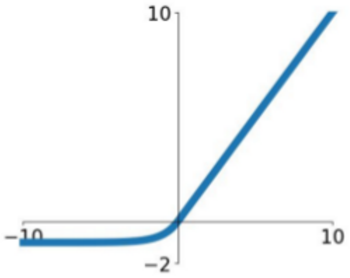


Maxout

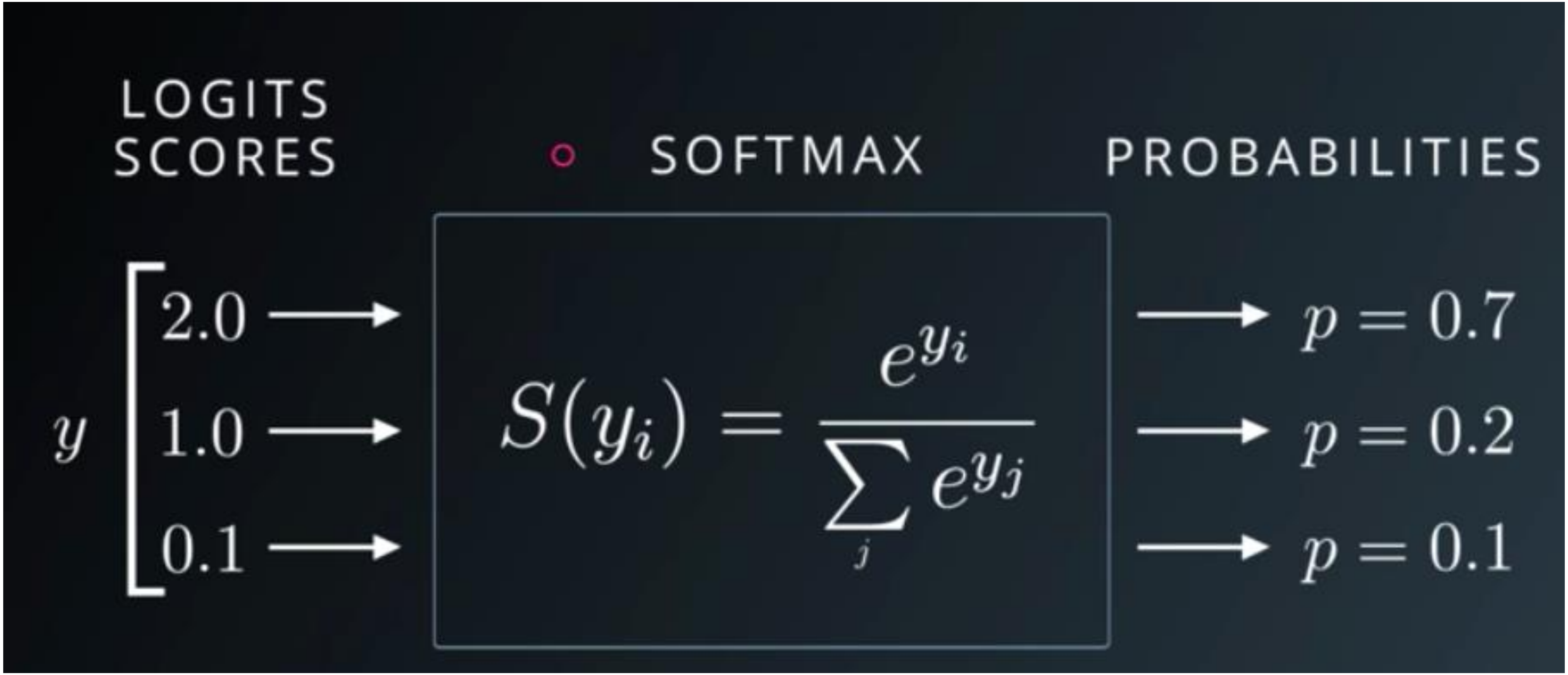
$\max(w_1^T x + b_1, w_2^T x + b_2)$

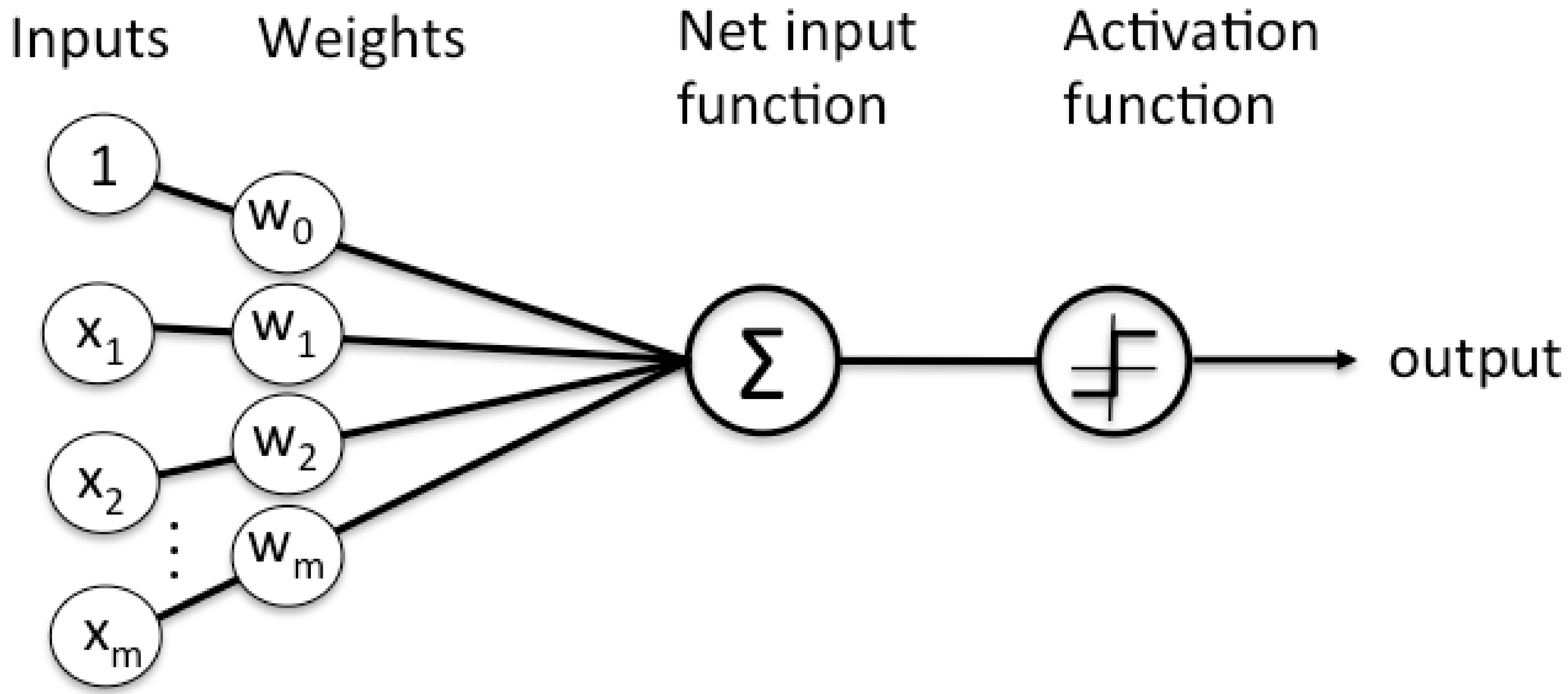
ELU

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



https://medium.com/@krishnakalyan3/introduction-to-exponential-linear-unit-d3e2904b366c

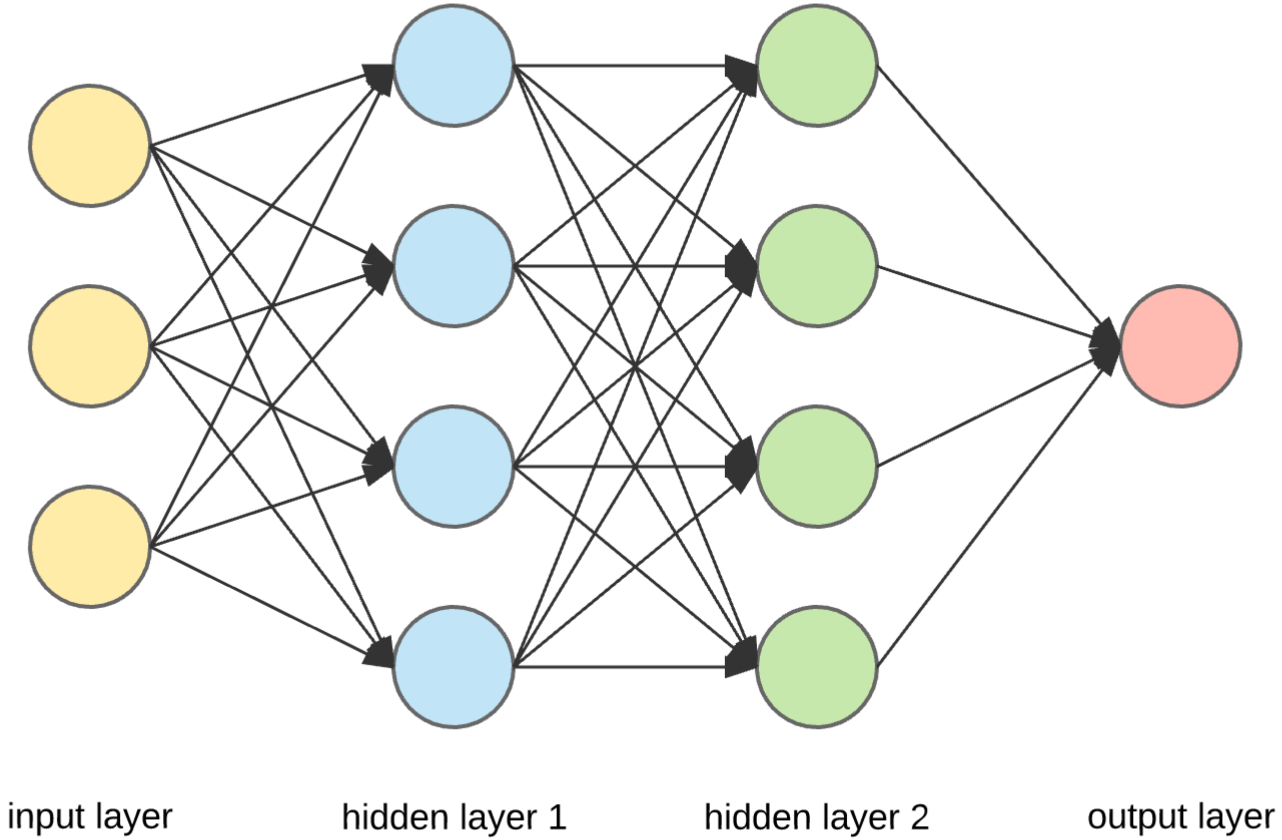

LOGITS
SCORES
SOFTMAX
PROBABILITIES
y
2.0
1.0
0.1
$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$
$p = 0.7$
$p = 0.2$
$p = 0.1$


Inputs
Weights
Net input function
Activation function
1
w_0
x_1
w_1
x_2
w_2
w_m
x_m
Σ
output

NEURON

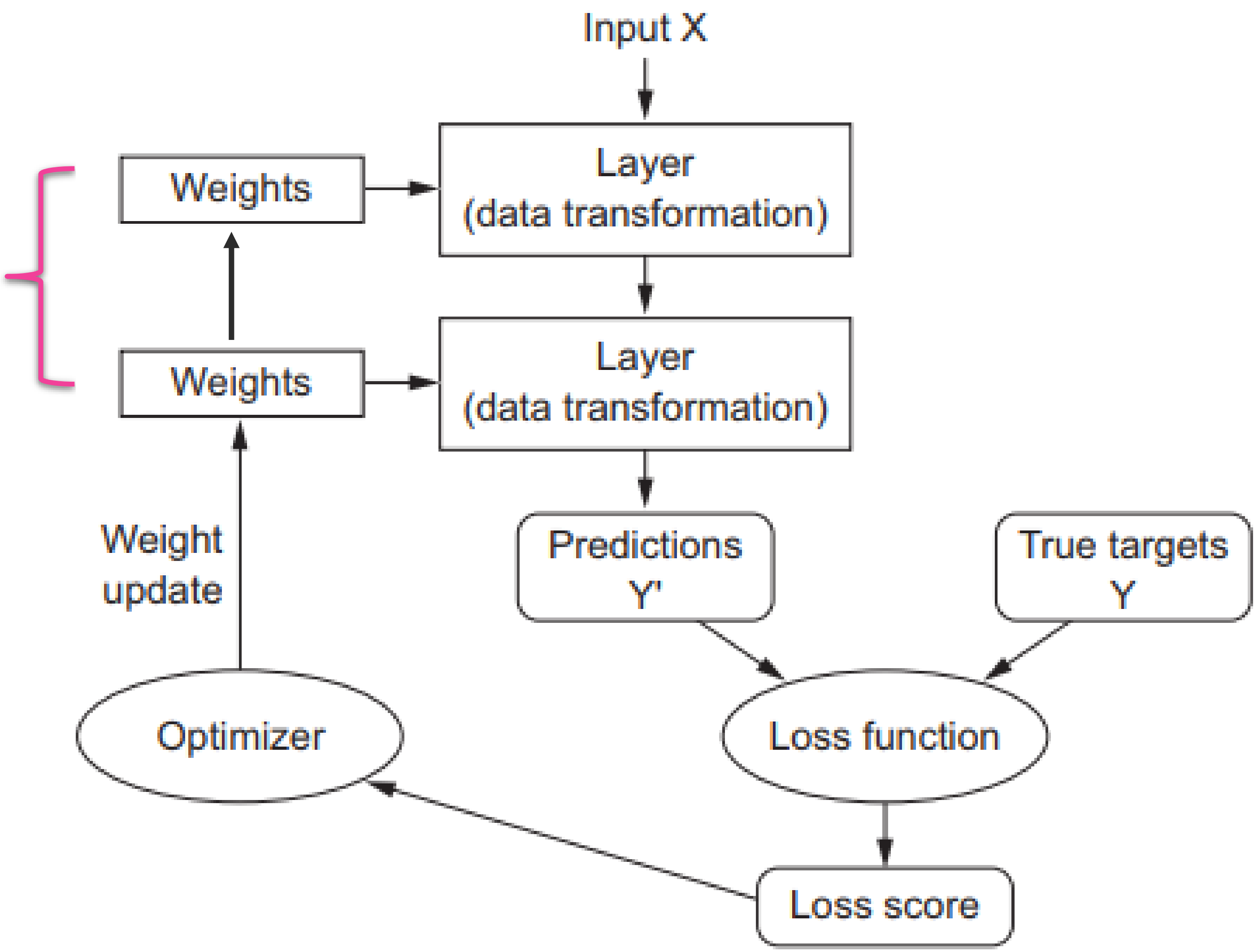
NEURAL NETWORK

- Input layer
- Hidden layer
- Output layer



DEEP LEARNING

- Goal: Finding the Rules--right values for the weights

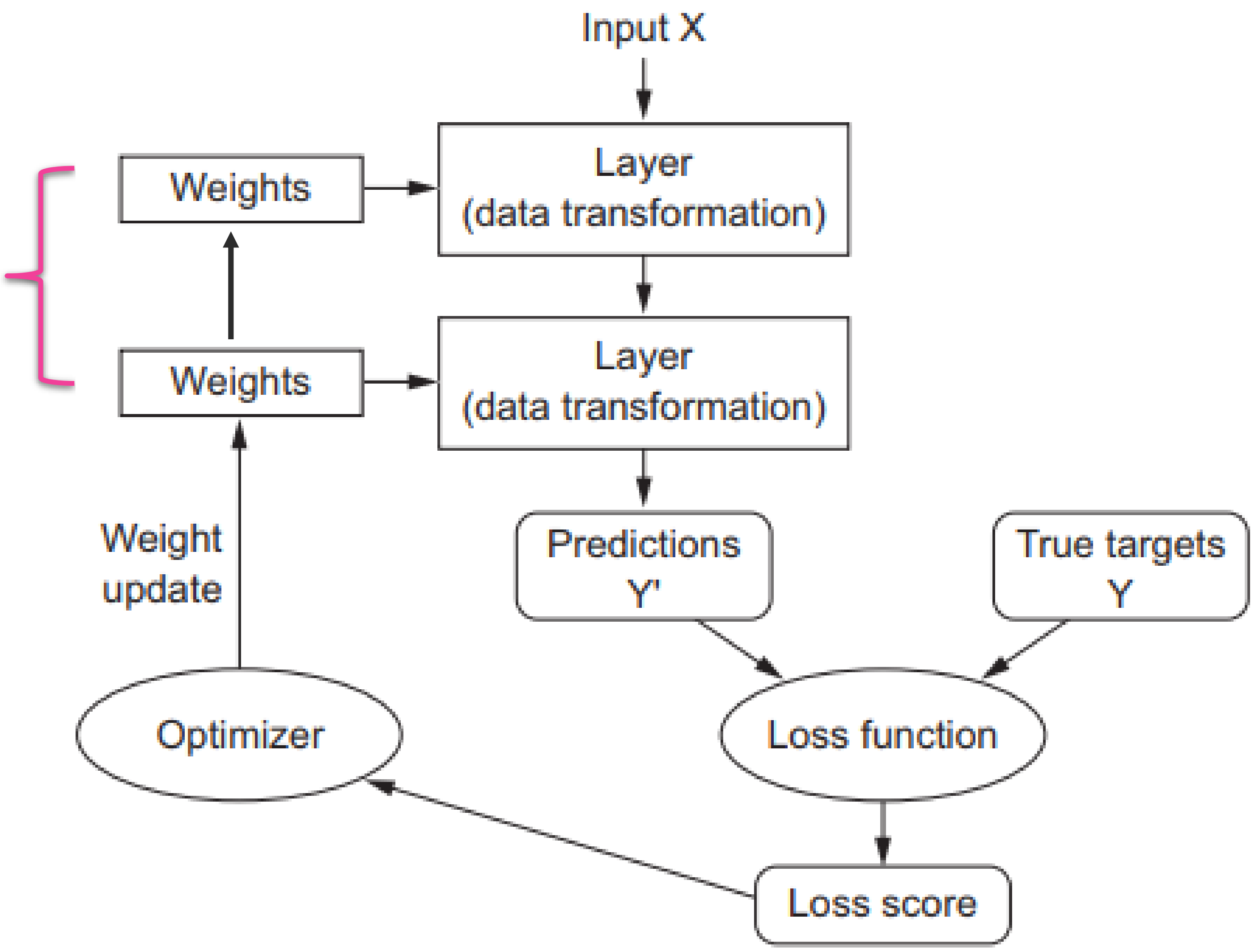


LOSS FUNCTIONS

- Mean Absolute Error (L1 Loss)
  - loss(y′,y)=|y′−y|
- Mean Square Error (L2 Loss)
  - loss(y′,y)=$(y'-y)^2$
- Binary Cross Entropy (BCE)
  - loss(y′,y)=−(ylog(y′)+(1−y)log(1−y′))
- Cross entropy
  - loss(y′,y)= $-\sum_{c=1}^{M} y_{o,c} \log(y'_{o,c})$
  - M - number of classes (dog, cat, fish)
  - log - the natural log
  - y - binary indicator (0 or 1) if class label c is the correct classification for observation o
  - y′ - predicted probability observation o is of class c

DEEP LEARNING

- Goal: Finding the Rules--right values for the weights

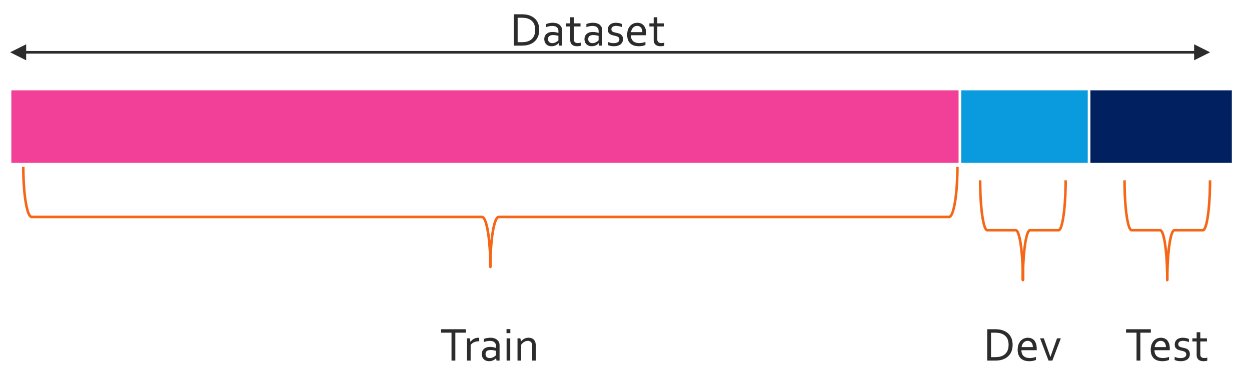


OPTIMIZERS

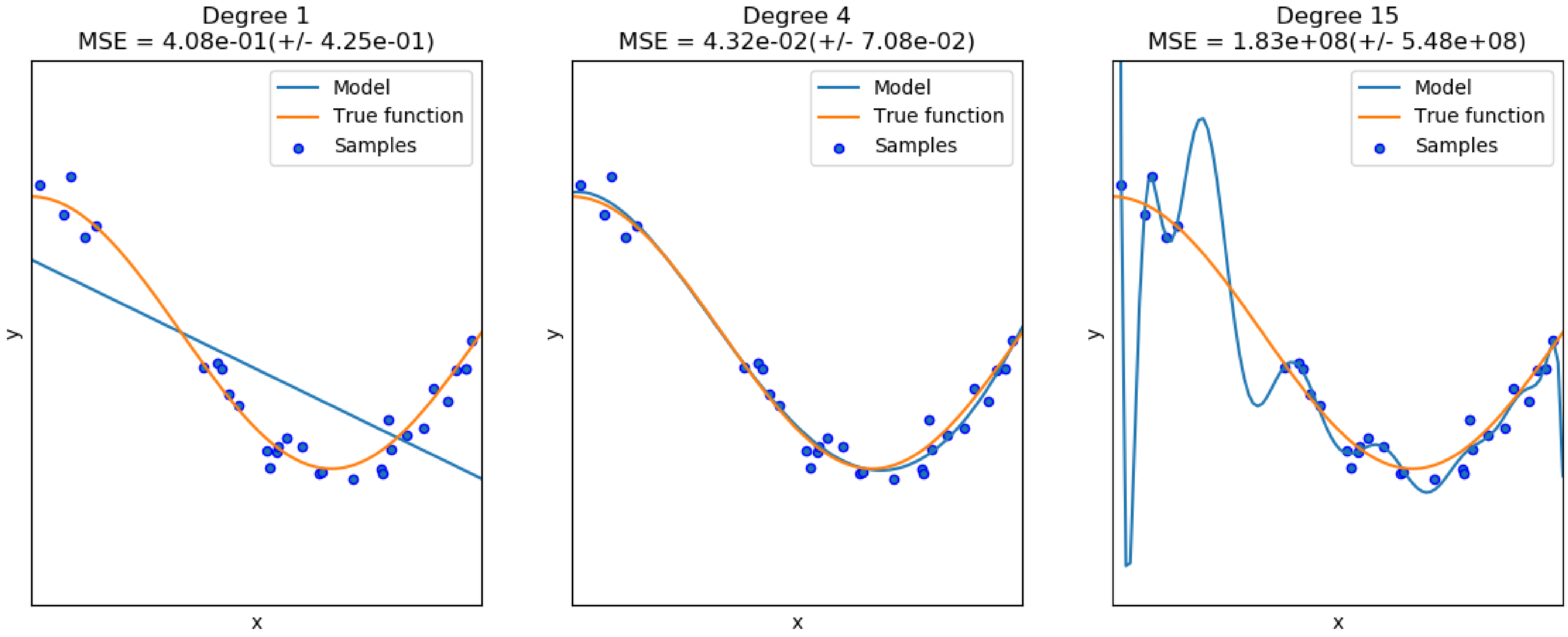
- We use optimization algorithms to train the neural network by optimizing the cost function ***J*.**
- The value of cost function ***J*** is the mean of the loss ***L*** between the predicted value ***y'*** and actual value ***y***.
- The value ***y'*** is obtained during the forward propagation step and makes use of the Weights ***W*** and biases ***b*** of the network.
- With the help of optimization algorithms, we minimize the value of Cost Function ***J*** by updating the values of the trainable parameters ***W*** and ***b***.

$$J(W,b) = \sum_{i=1}^{m} L(y'^{i}, y^{i})$$

EVALUATION MACHINE LEARNING MODELS



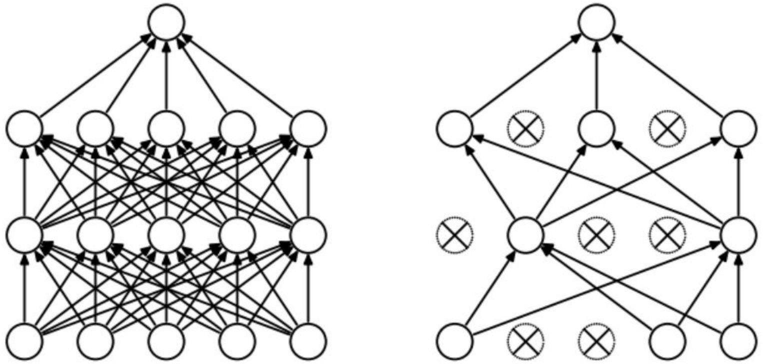
OVERFITTING AND UNDERFITTING



https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html

PREVENT OVERFITTING

- Get more training data
- Reducing the network size
- Regularization
  - L1 regularization—The cost added is proportional to the absolute value of the weight coefficients
  - L2 regularization—The cost added is proportional to the square of the value of the weight coefficients
- Dropout



(a) Standard Neural Net (b) After applying dropout.

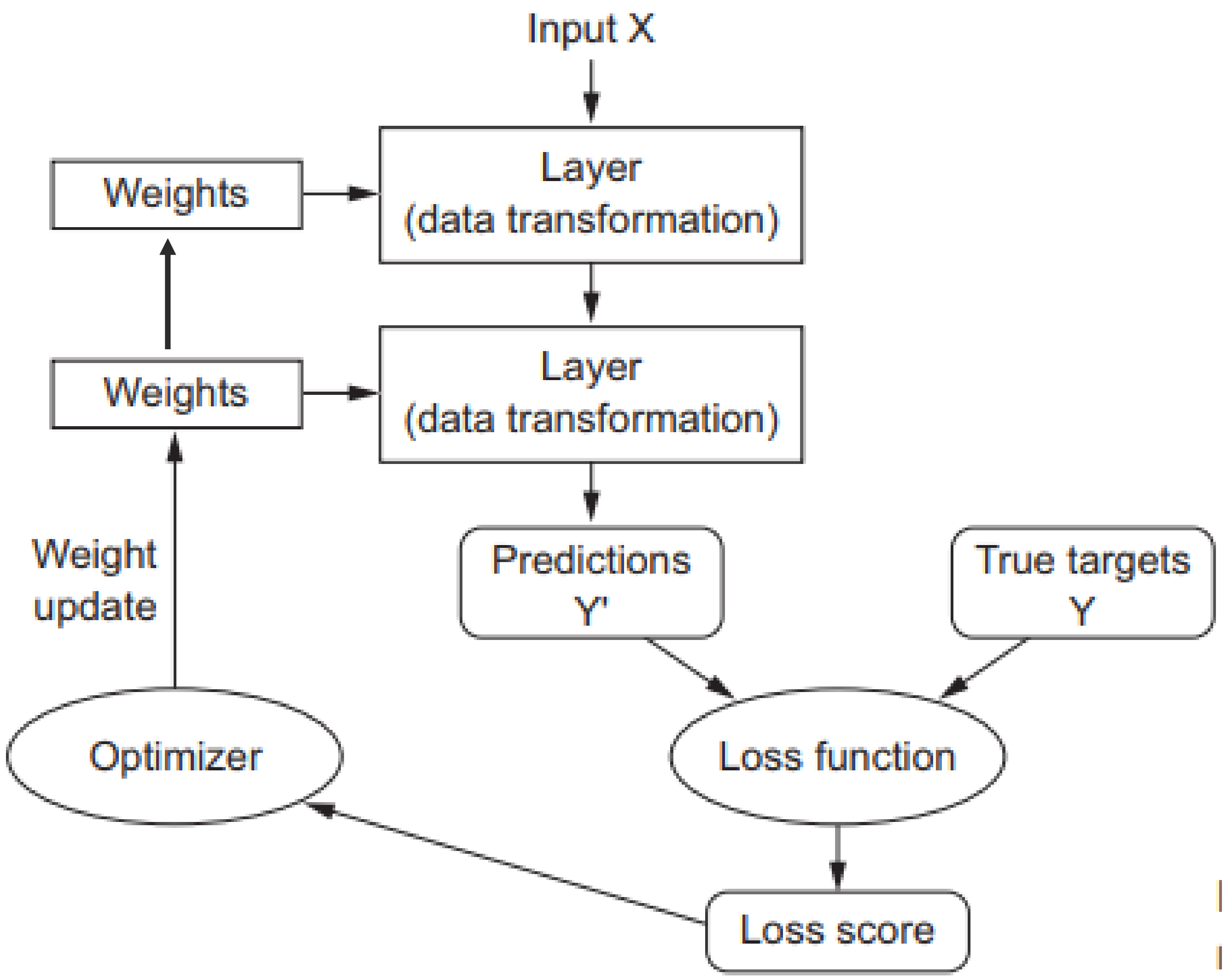
https://miro.medium.com/max/700/1*iWQzxhVlvadk6VAJjsgXgg.png

AVOID UNDERFITTING

- Increase model complexity
- Reduce the regularization parameters

ANATOMY OF A NEURAL NETWORK

- Layers, which are combined into a network (or model)
- The input data and corresponding targets
- The loss function, which defines the feedback signal used for learning
- The optimizer, which determines how learning proceeds



REFERENCES

- Francois Chollet. 2017. Deep Learning with Python (1st ed.). Manning Publications Co., Greenwich, CT, USA.
- https://ml-cheatsheet.readthedocs.io/en/latest/optimizers.html
- https://pytorch.org/tutorials/
- https://keras.io/
- https://www.tensorflow.org/guide/keras