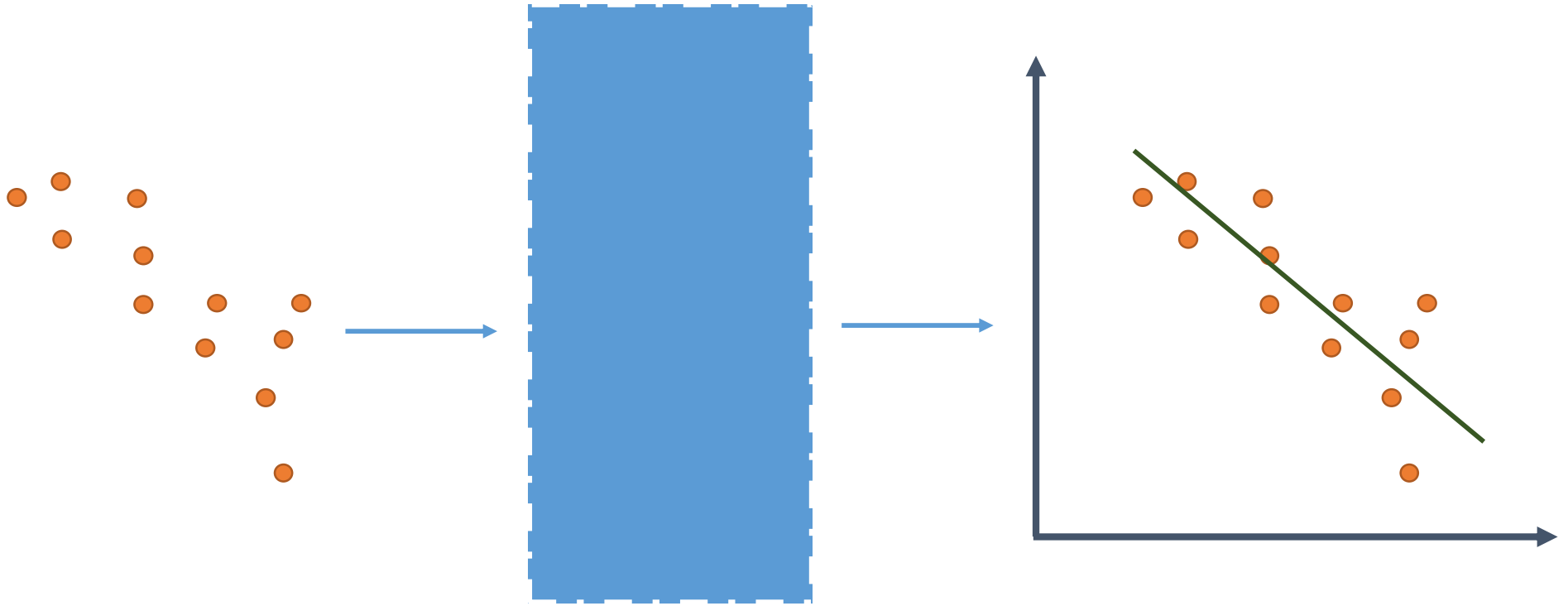
# Activation & Loss functions

Functions to be used in case of classification problem

# Regression Problem

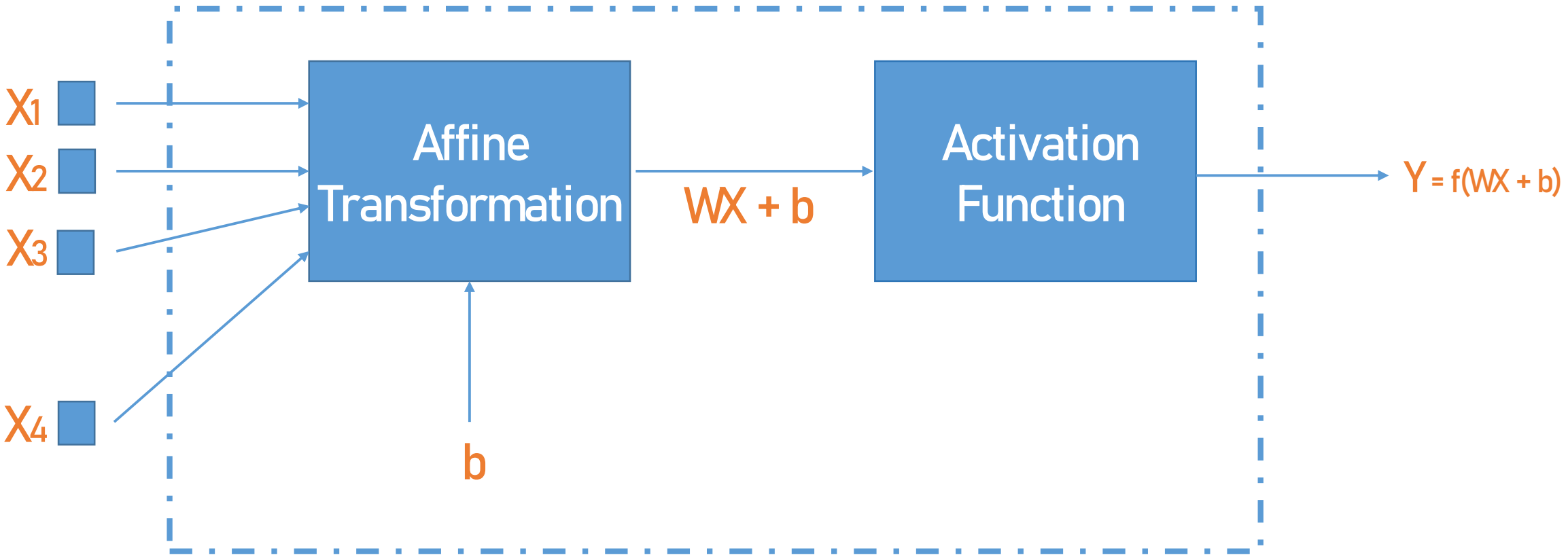


Points in 2-D

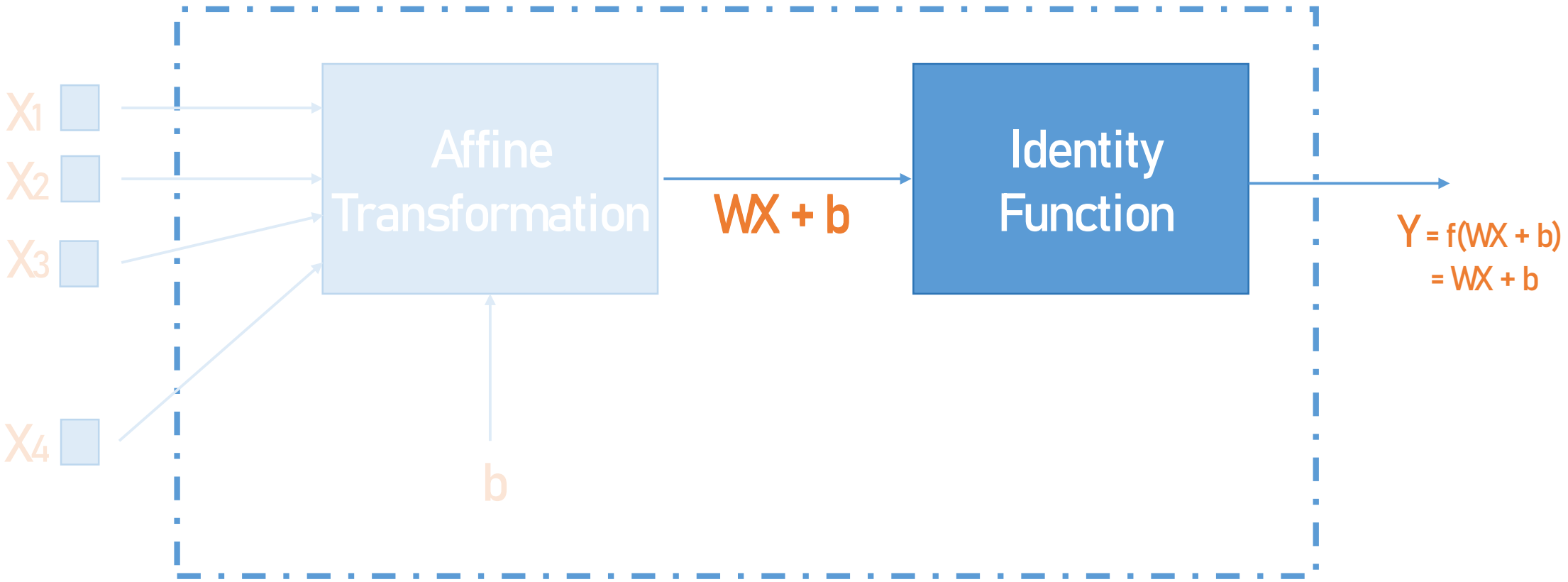
Single Neuron

Regression

# Neuron Mathematical Functions

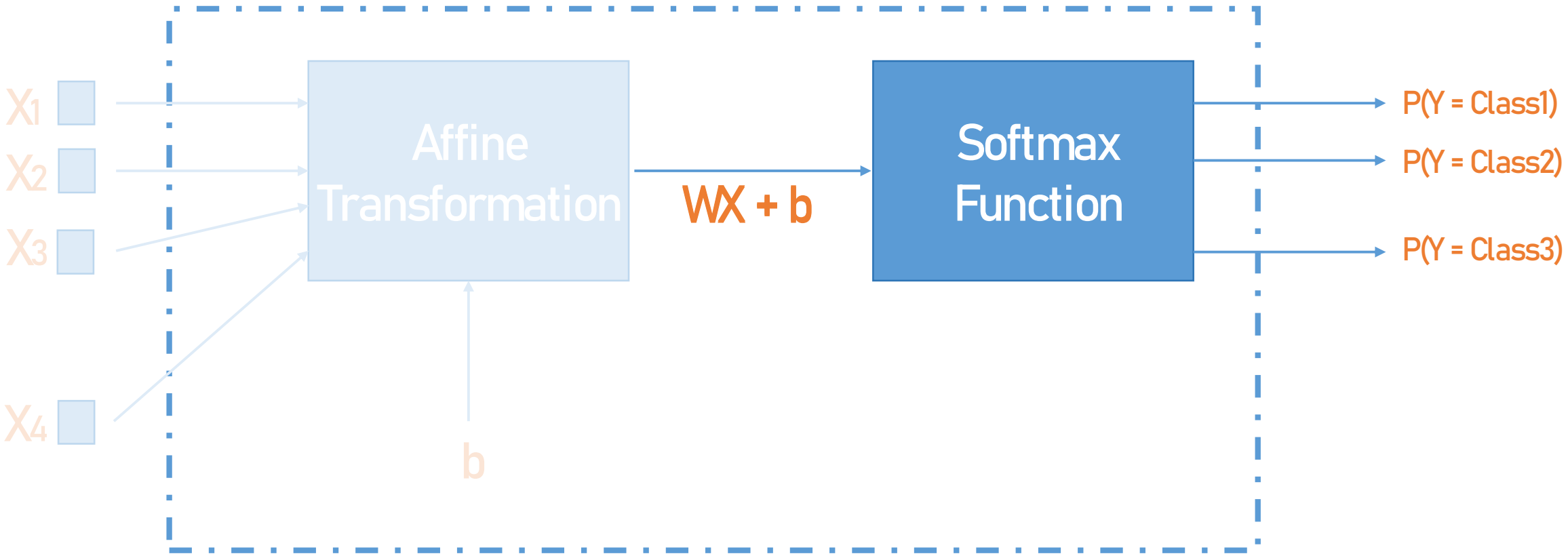



# Neuron Mathematical Functions



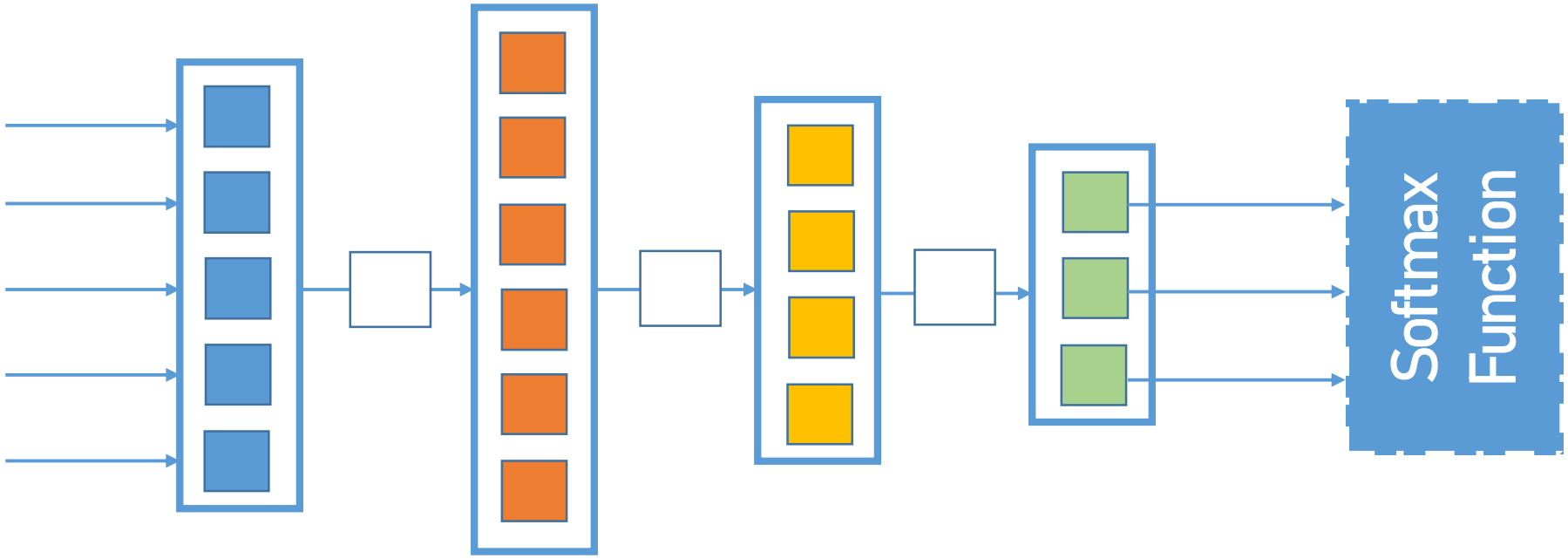

Activation function is an Identity Function

# Neuron Mathematical Functions

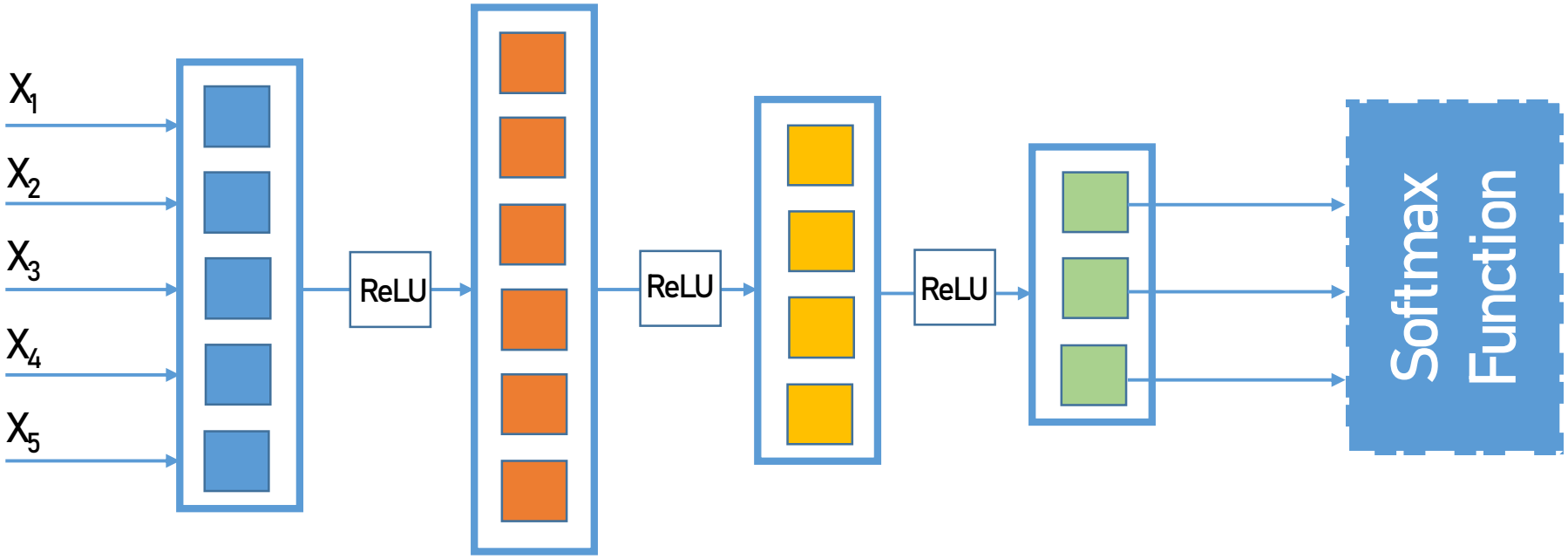



Activation function is an Identity Function

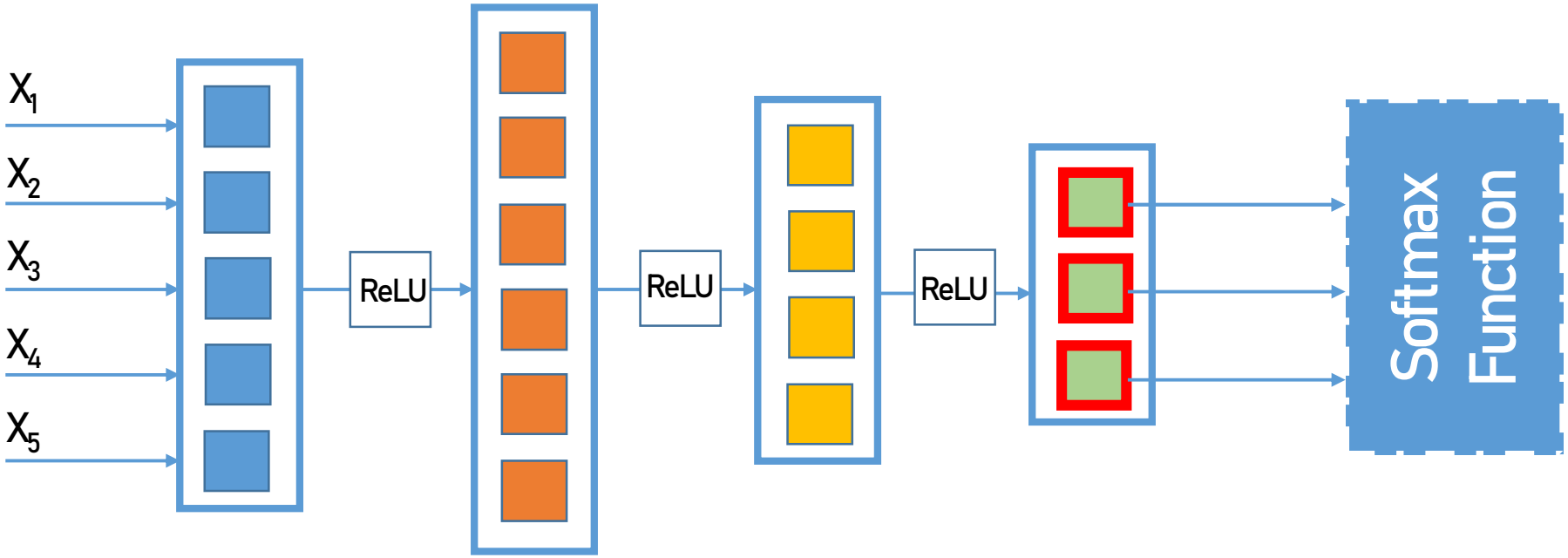
# Neural Network with Softmax




# Neural Network with Softmax




# Neural Network with Softmax




Number of neurons in last layer = number of classes

# Softmax Function

$$Z_N^1 = W_N X_N + b_N$$

$$Z_N^2 = W_N X_N + b_N$$

$$Z_N^3 = W_N X_N + b_N$$

# Softmax Function

$Z_N^1$

$Z_N^2$

$Z_N^3$

# Softmax Function

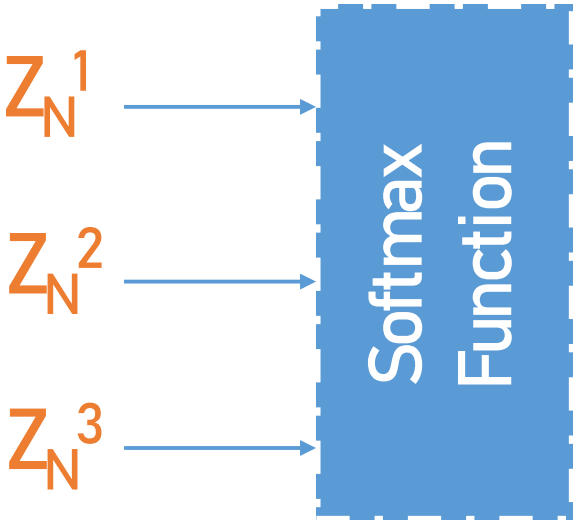
$Z_N^1$

$Z_N^2$

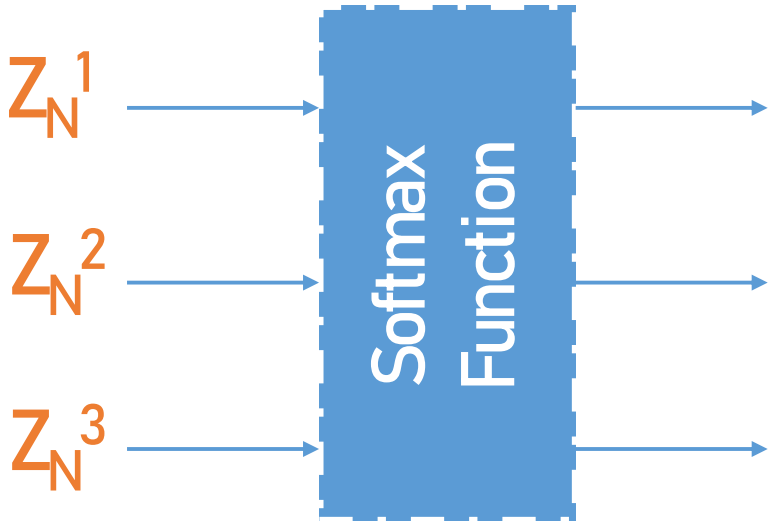
$Z_N^3$

Softmax Function

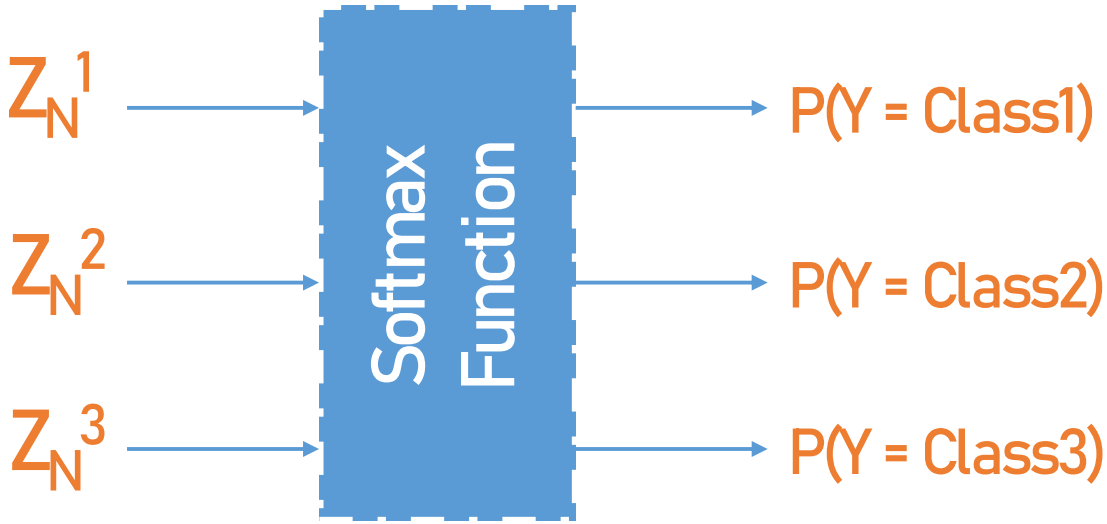
# Softmax Function



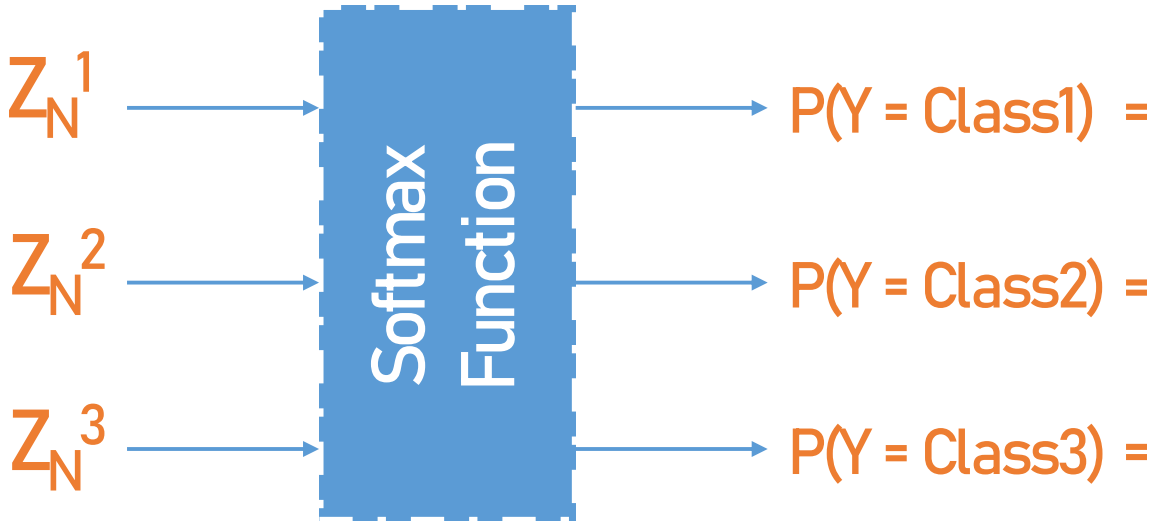

# Softmax Function



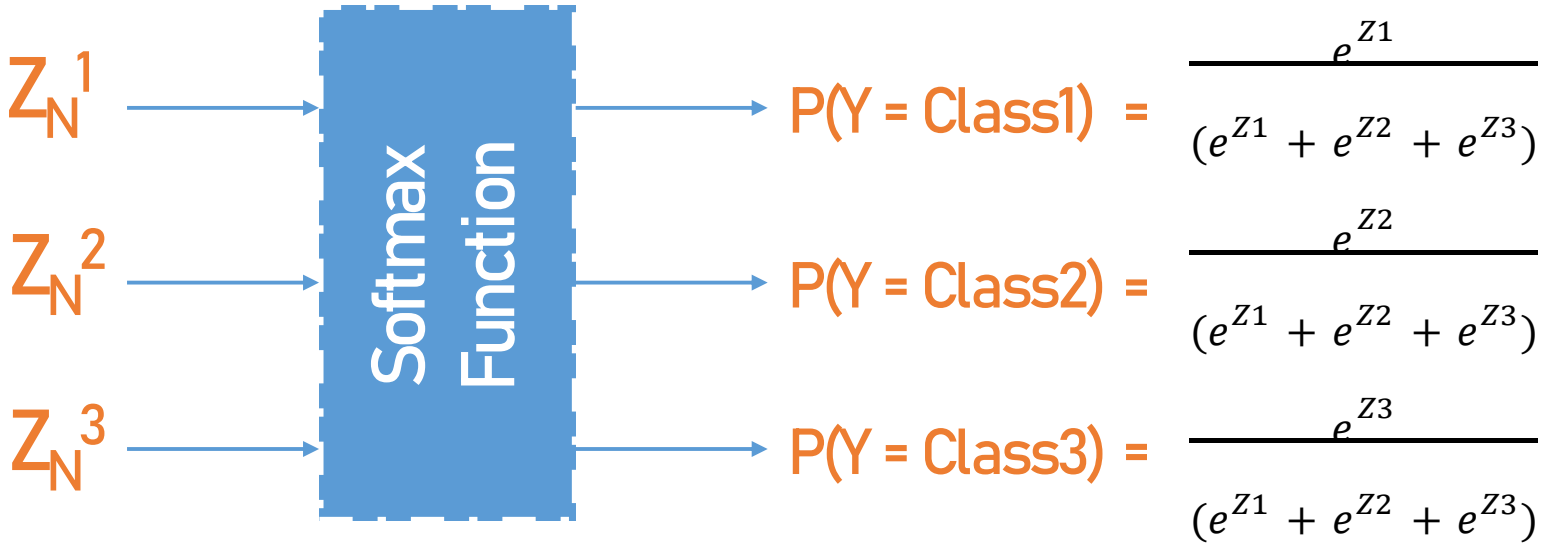

# Softmax Function




# Softmax Function




# Softmax Function

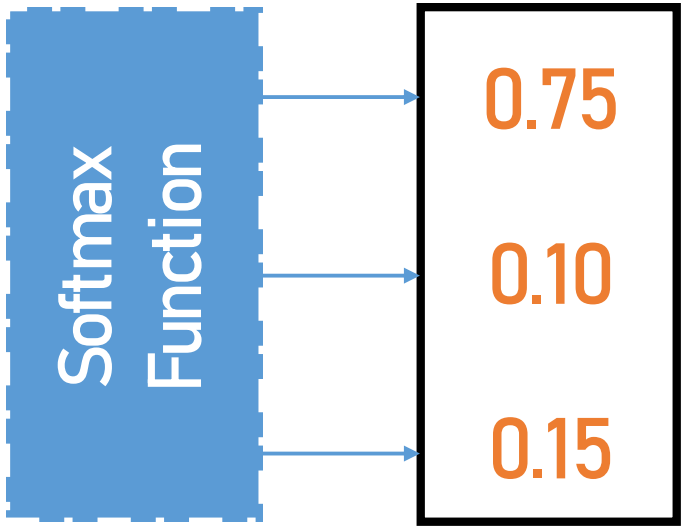



The class having highest probability is selected

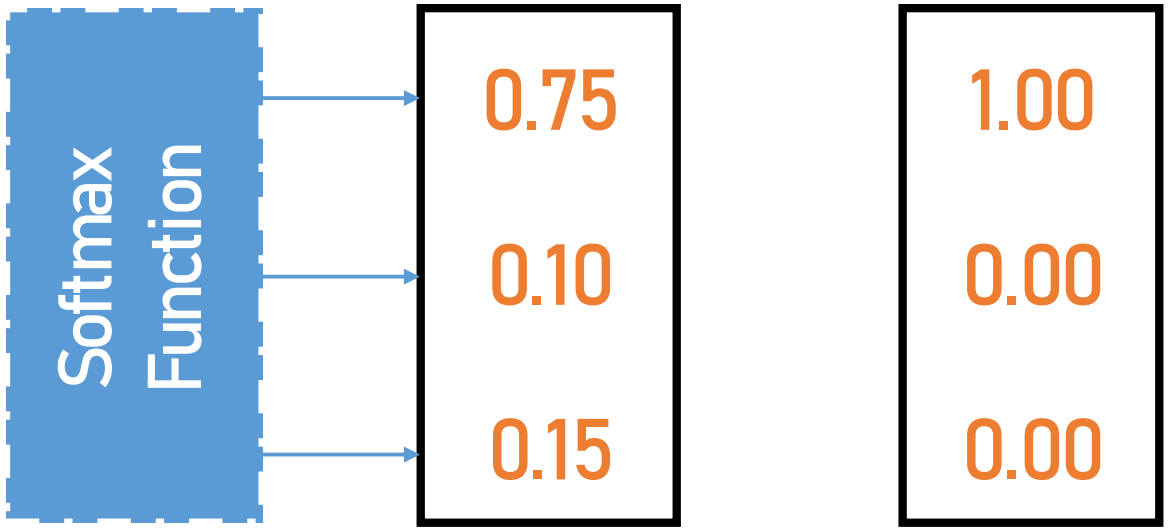
For classification the loss function we use is called cross entropy loss

Cross entropy is a metric that signifies how different two probability distributions are

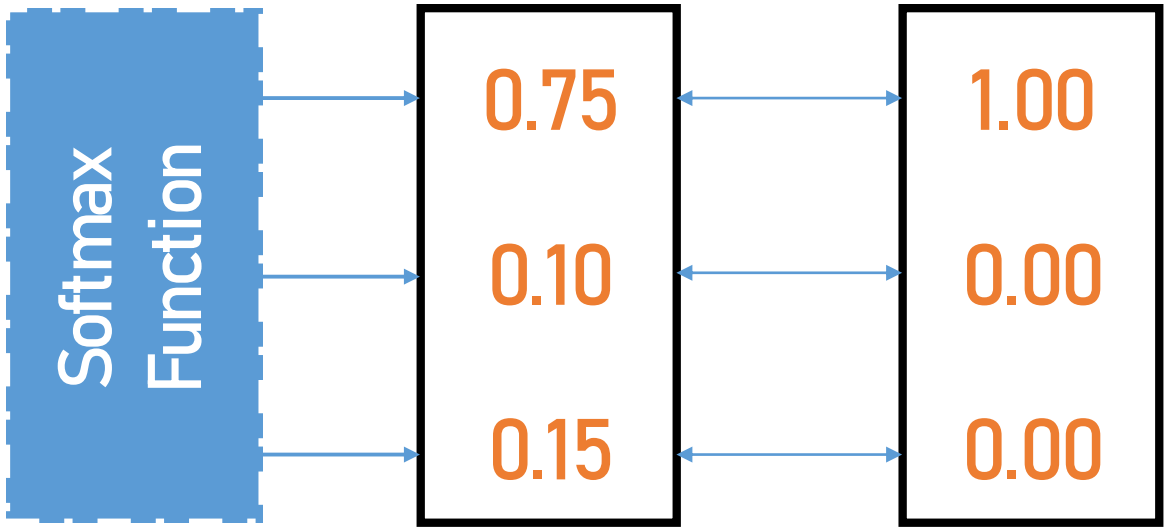
# Softmax Function



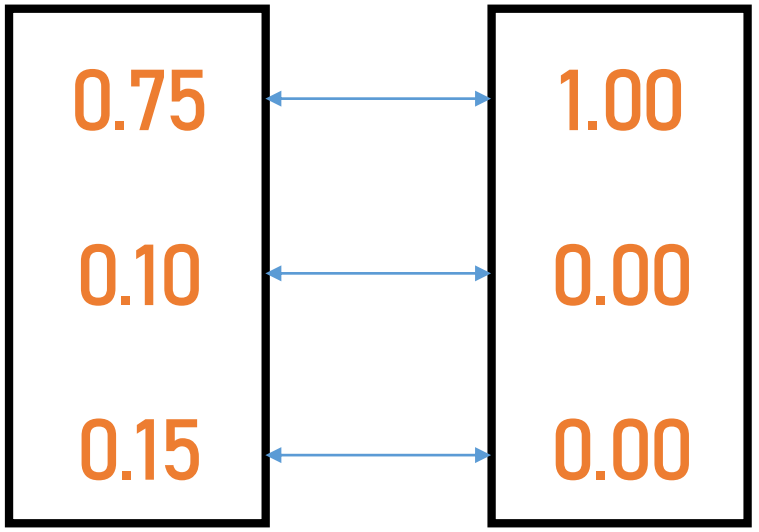

# Softmax Function



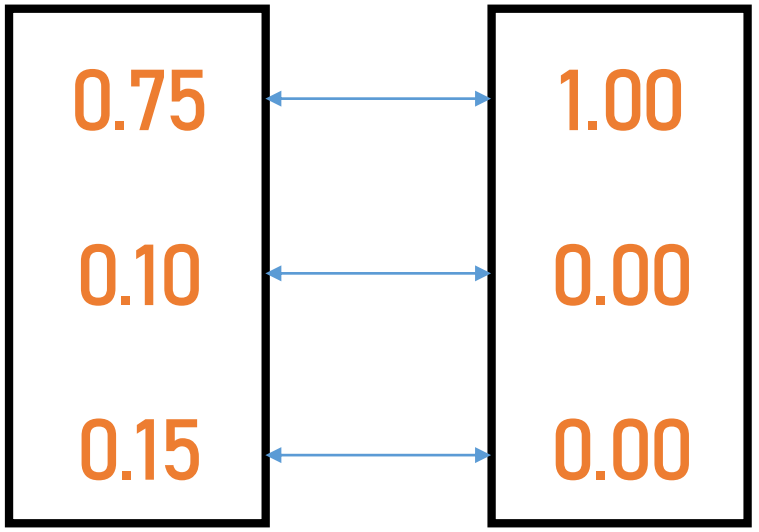

# Softmax Function




# Softmax Function

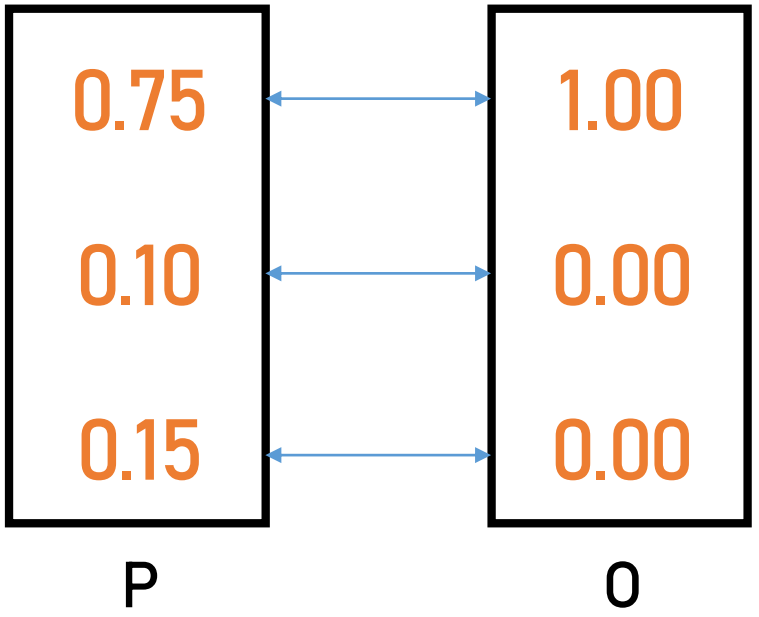



# Softmax Function




$$CE(O, P) = -\sum_{i=0}^{n} O_i \log P_i$$

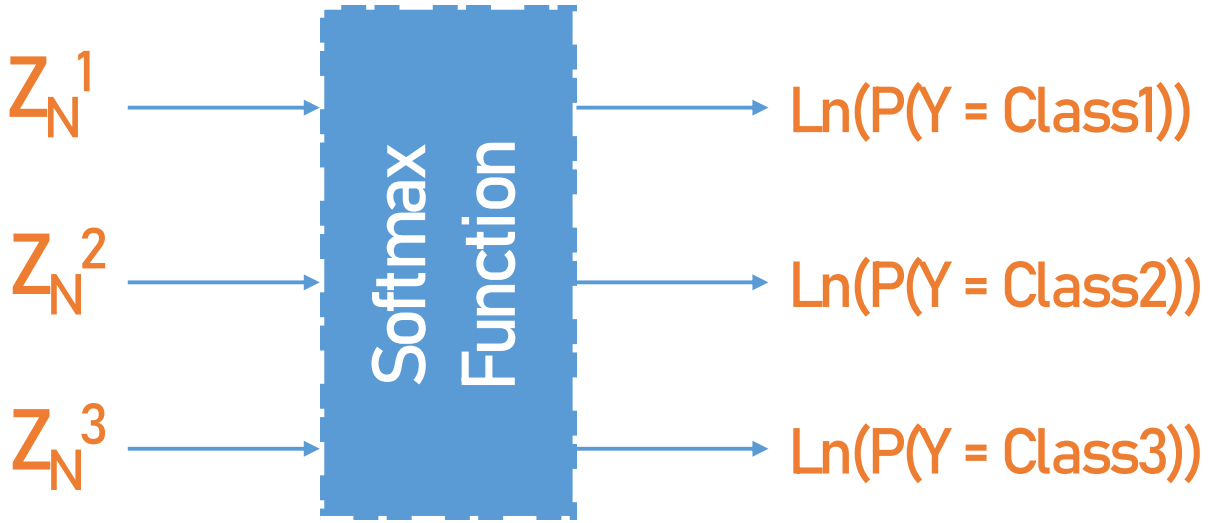
# Softmax Function




$$CE(O, P) = -\sum_{i=0}^{n} O_i \log P_i$$

In PyTorch examples we will mostly use LogSoftmax activation function along with NLLLoss

# LogSoftmax Function




LogSoftmax is Log of Softmax

Softmax + Cross Entropy ⇔ LogSoftmax + NLLLoss