Machine Learning

Neural Network

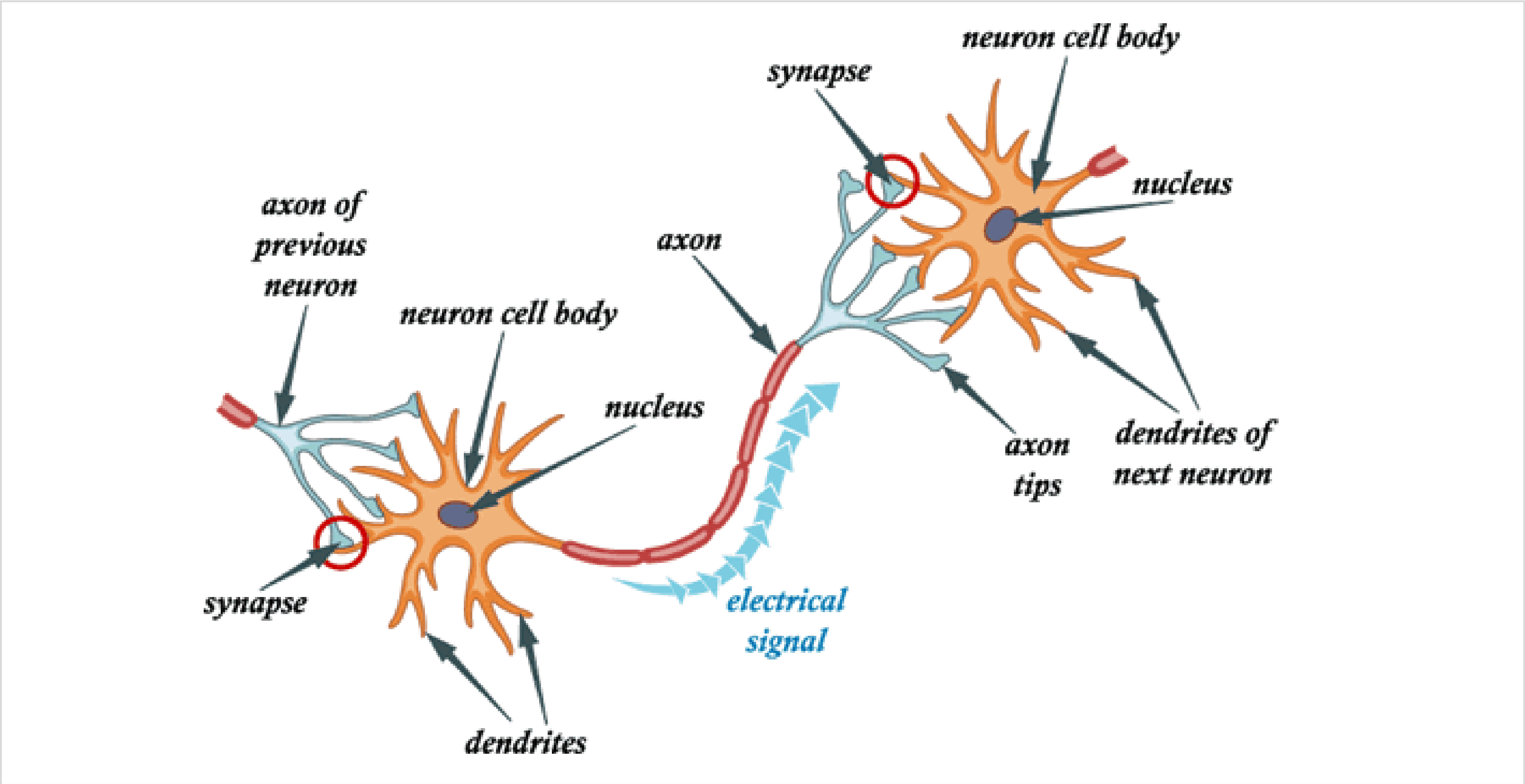
김선녕(sykim.lecture@gmail.com)

개요

퍼셉트론

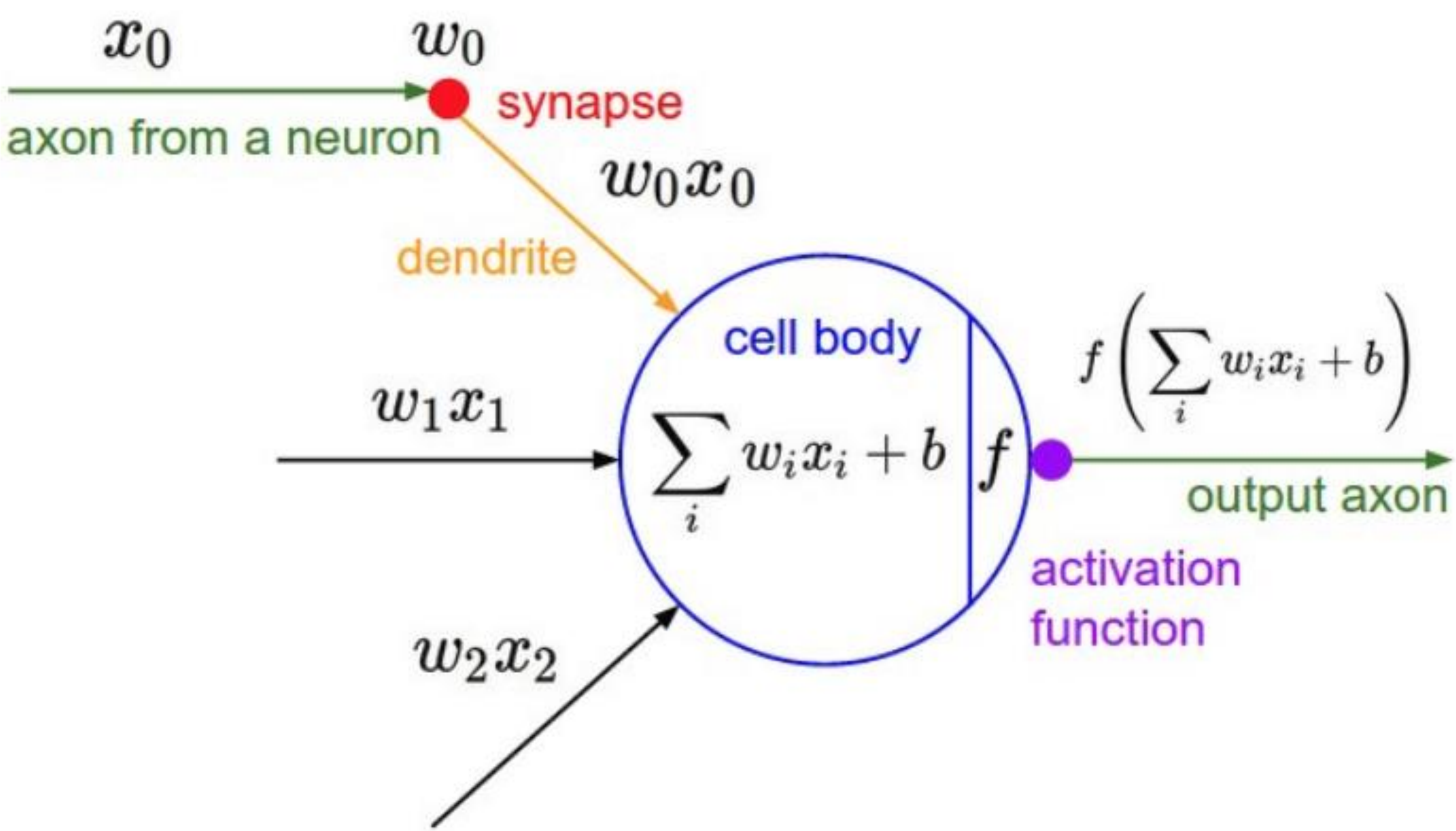
Biological neuron

- 인간의 뇌는 1000억 개가 넘는 뉴런이 100조 개 이상의 시냅스 (*synapse*) 를 통해 병렬적으로 연결되어 있다고 한다.
- 뉴런은 수상돌기(*dendrite*)를 통해 다른 뉴런에서 입력 신호를 받아서 축색돌기(*axon*)를 통해 다른 뉴런으로 신호를 내보낸다.
- 시냅스(*synapse*) : 뉴런과 뉴런을 연결하는 역할



https://medium.com/autonomous-agents/mathematical-foundation-for-activation-functions-in-artificial-neural-networks-a51c9dd7c089

Mathematical Model



http://cs231n.github.io/neural-networks-1/

- 수백만 개의 이미지 분류 : 구글 이미지
- 음성 인식 서비스의 성능 효율 증대 : 시리
- 매일 수억 명에 이르는 사용자에게 가장 좋은 비디오 추천 : 유튜브
- 바둑 세계 챔피언을 이기기 위해 수백만 개의 기보를 익히고 자신과 게임하면서 학습 : 딥마인드의 알파고
- 매우 복잡한 대규모 머신러닝 문제를 다루는 데 적합

개요

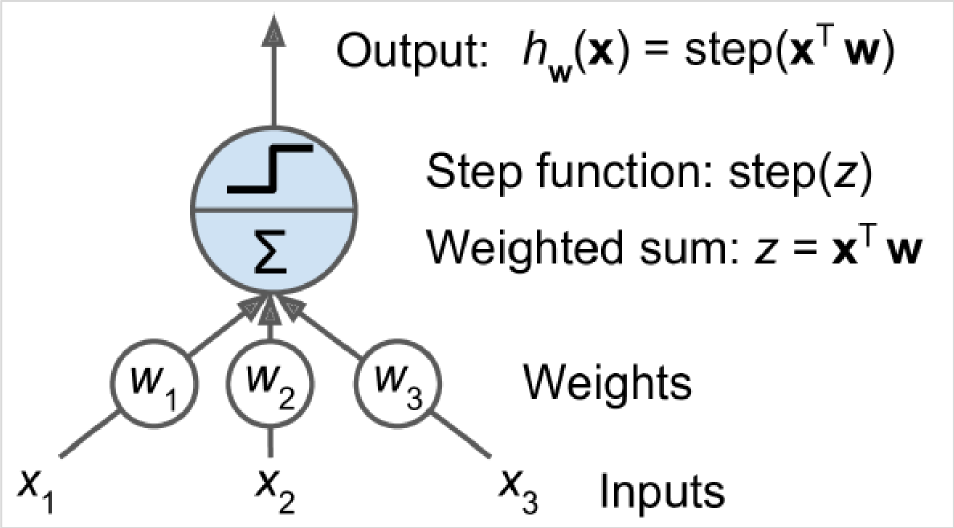
퍼셉트론

퍼셉트론(Perceptron)

- 1957년 프랭크 로젠블랫(Frank Rosenblatt)에 의해 고안된 알고리즘
- TLU(threshold logic unit) 혹은 LTU(linear threshold unit)이라고도 불린다
- 다수의 신호($Input: x_1, x_2 \cdots$)을 입력 받아서 계단함수를 적용하여 결과 신호($Output$: y)를 출력

$$w_1x_1 + w_2x_2 + \cdots + w_nx_n = \sum_{i=1}^{n} w_ix_i = w^Tx$$

- 가중치($weight$) : 특징(feature)이 레이블(label)의 예측에 끼치는 영향도. 값이 클수록 예측에 미치는 영향이 크다. 학습이 진행되면서 값이 변동
- 임계값($threshold$) : 뉴런에서 보낸 신호의 총합이 정해진 한계치. θ로 표시

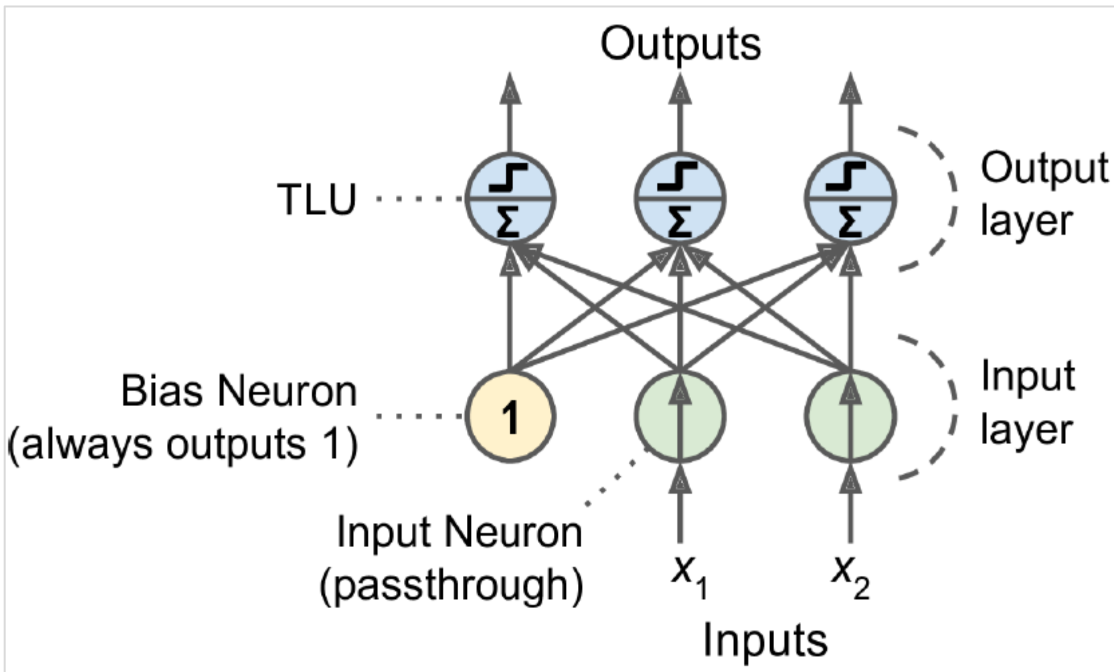


http://www.edubilla.com/inventor/frank-rosenblatt/

출력 계산

$$h_{w,b}(X) = \phi(WX + b)$$

– ϕ : 활성화 함수($activation\ function$). TLU인 경우 계단함수($step\ function$)



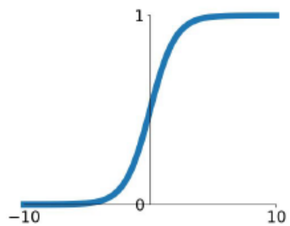
입력뉴런 두개, 편향 뉴런 한 개, 출력 뉴런 세개로 구성된 퍼셉트론 구조

활성화 함수

- 입력받은 신호를 이를 적절한 처리를 하여 다음 뉴런(층)으로 출력하는 함수

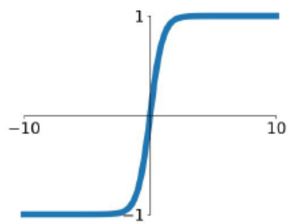
Sigmoid

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



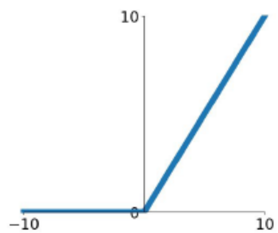
tanh

$\tanh(x)$



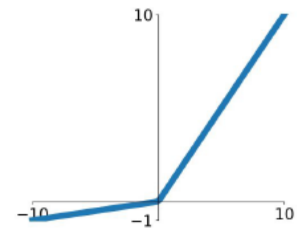
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

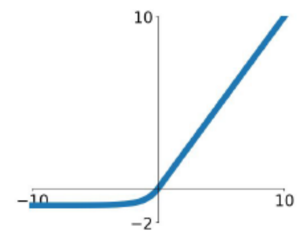


Maxout

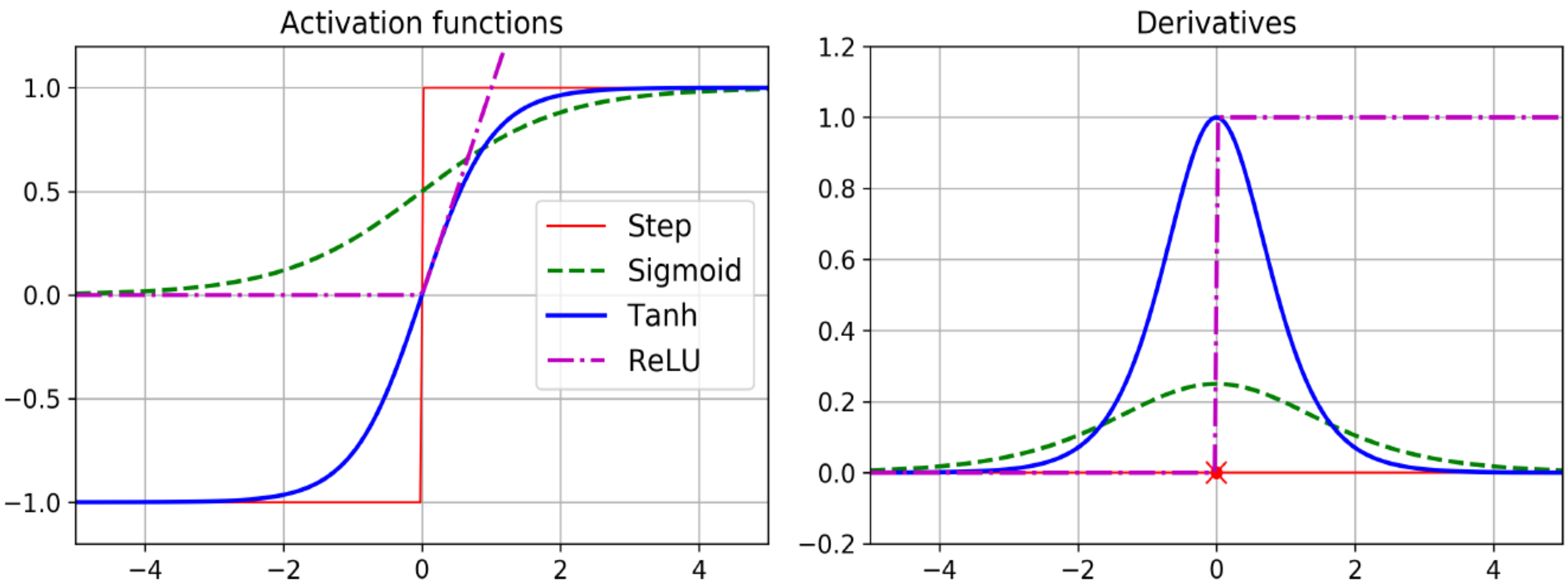
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

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



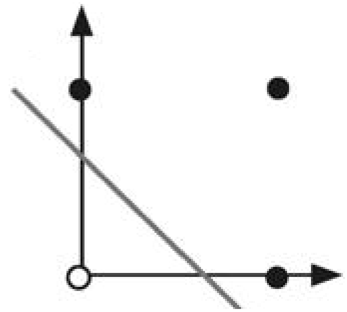
활성화 함수와 미분

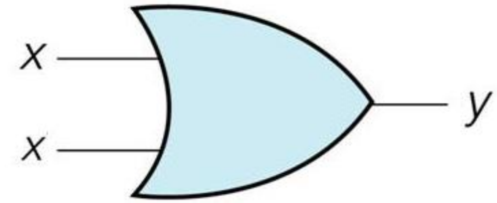


Activation functions
1.0
0.5
0.0
−0.5
−1.0
−4
−2
0
2
4
Step
Sigmoid
Tanh
ReLU
Derivatives
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−4
−2
0
2
4

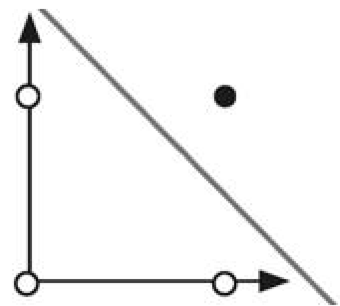
퍼셉트론 - XOR

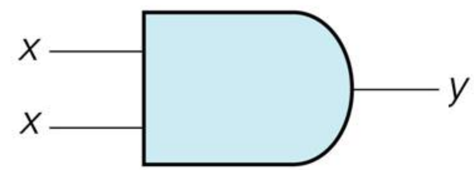
x	x	y
0	0	0
0	1	0
1	0	0
1	1	1



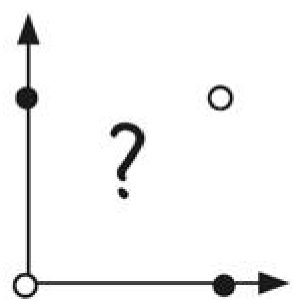


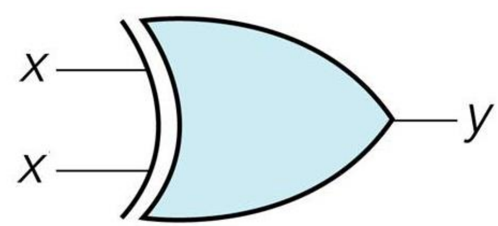
x	x	y
0	0	0
0	1	1
1	0	1
1	1	1





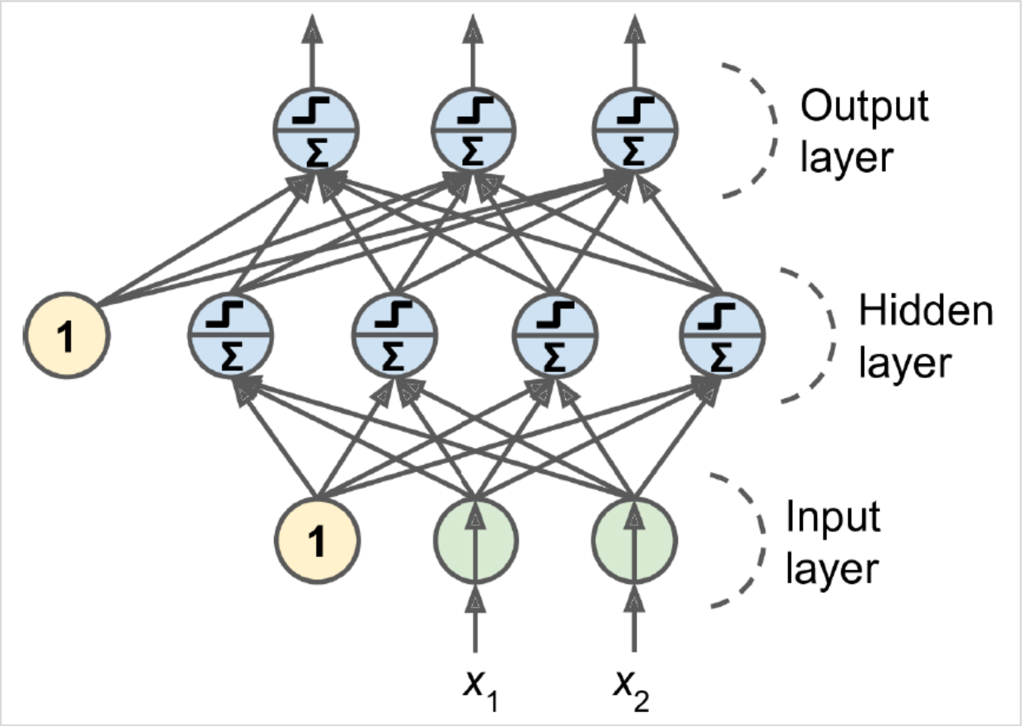
x	x	y
0	0	0
0	1	1
1	0	1
1	1	0





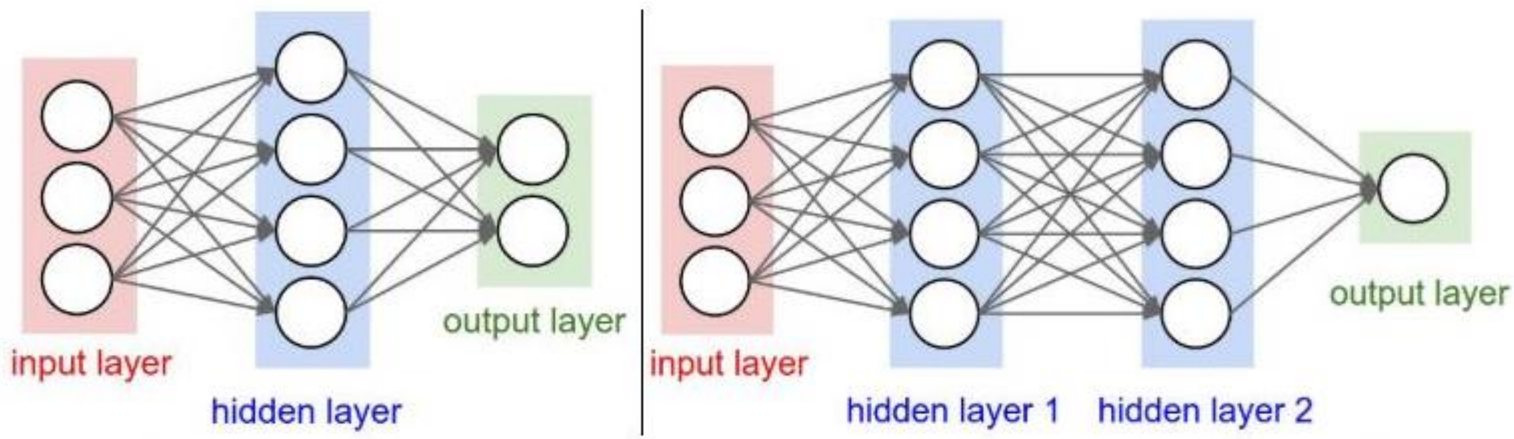
다층 퍼셉트론(MLP, Multilayer Perceptron)

- 입력층(input layer), 은닉층(hidden layer), 출력층(output layer)
- 출력층을 제외하고 모든 층은 편향 뉴런 포함
- 순전파 신경망(feedforward neural network) : 신호가 한방향으로 흐른다
- 심층 신경망(deep neural network, DNN) : 은닉층 여러 개인 망, 딥러닝



신경망 (Neural Networks) 구조

- 신경망의 크기를 측정하는 척도 : 뉴런의 수 혹은 parameter의 수
  - parameter : 뉴런과 뉴런의 연결된 부분의 weight or bias



Left: A 2-layer Neural Network (one hidden layer of 4 neurons (or units) and one output layer with 2 neurons), and three inputs. **Right:** A 3-layer neural network with three inputs, two hidden layers of 4 neurons each and one output layer. Notice that in both cases there are connections (synapses) between neurons across layers, but not within a layer.

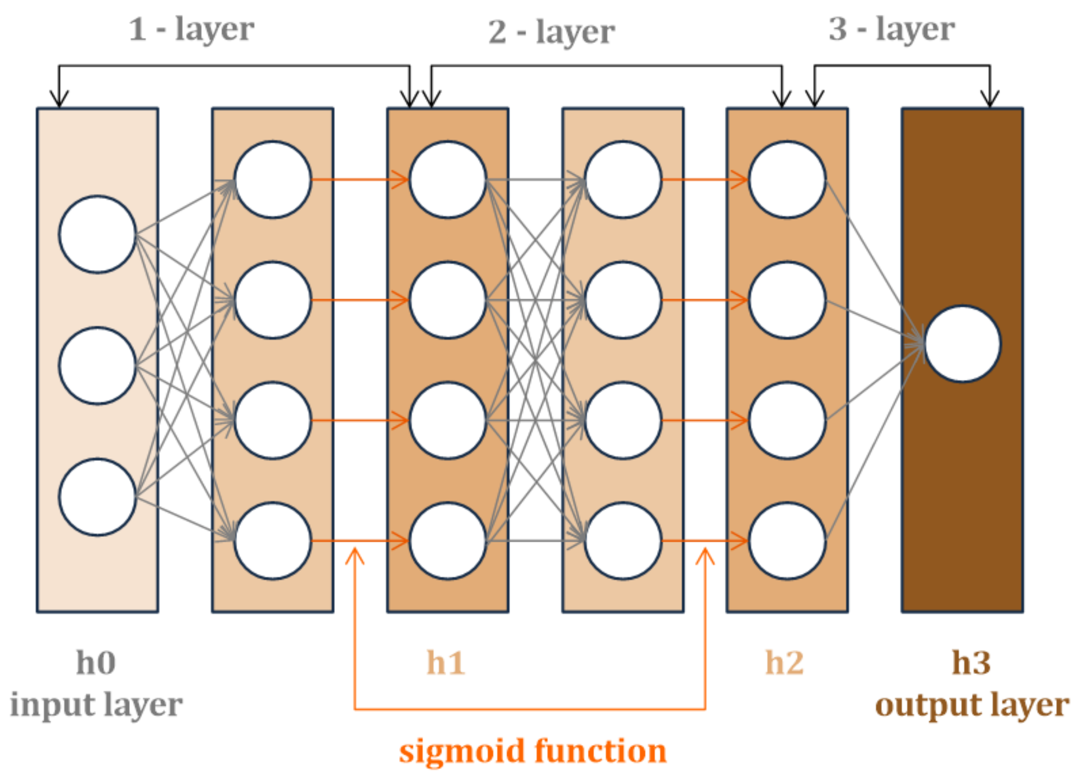
- Left
  - 4+2 = 6개의 뉴런. [3x4] + [4x2] = 20 개의 weights와 4+2 = 6개의 biases. 총 26개의 parameters
- Right
  - 4+4+1 = 9개의 뉴런. [3x4] + [4x4]+[4x1] = 32개의 weights와 4+4+1의 biases. 총 41개의 parameters

- 대략 10~20개의 층이 있는 신경망의 parameters의 수는?

http://cs231n.github.io/neural-networks-1/

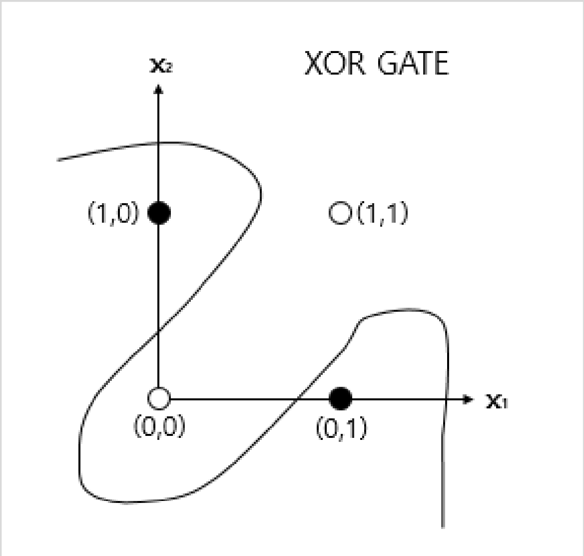
feed-forward computation

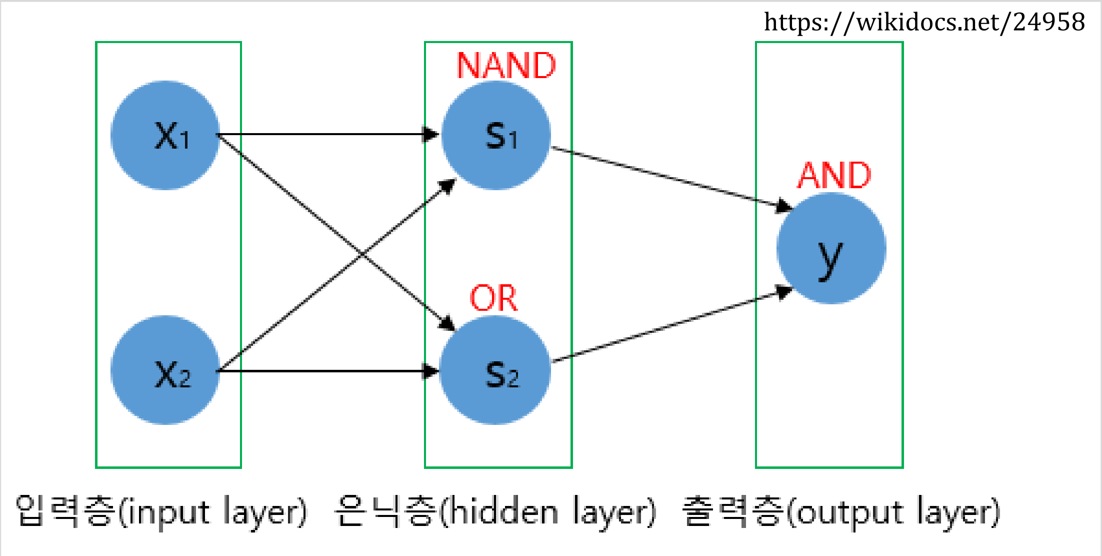
- **W1** = 1층 [4 x 3] 크기를 가지는 weight
- **b1** = 1층 bias vector
- **W2** = 2층 [4 x 4] 크기를 가지는 weight
- **b2** = 2층 bias vector
- **W3** = 3층 [1 x 4] 크기를 가지는 weight 라고 하면

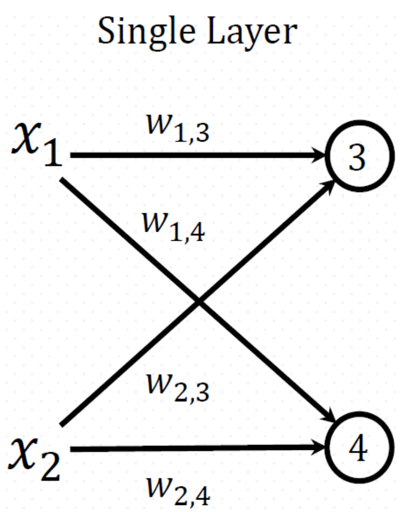


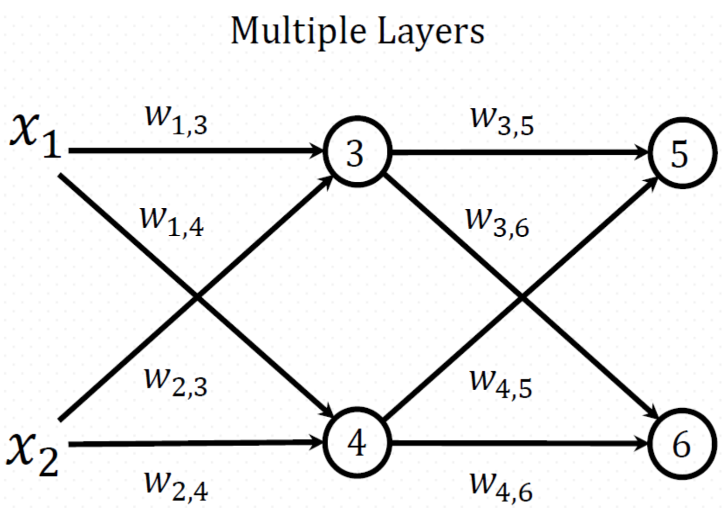
http://cs231n.github.io/neural-networks-1/

XOR


XOR GATE
x2
x1
(1,0)
(1,1)
(0,0)
(0,1)


https://wikidocs.net/24958
NAND
AND
OR
x1
x2
s1
s2
y
입력층(input layer)
은닉층(hidden layer)
출력층(output layer)


Single Layer
x_1
x_2
$w_{1,3}$
$w_{1,4}$
$w_{2,3}$
$w_{2,4}$
3
4


Multiple Layers
x_1
x_2
$w_{1,3}$
$w_{1,4}$
$w_{2,3}$
$w_{2,4}$
$w_{3,5}$
$w_{3,6}$
$w_{4,5}$
$w_{4,6}$
3
4
5
6

- $\hat{y} = Wx + b$

$$X \xrightarrow{W} \Sigma \xrightarrow{z} \sigma(z) \longrightarrow \hat{y} = \begin{cases} 0 \\ 1 \end{cases}$$

- $W = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, b = -8$

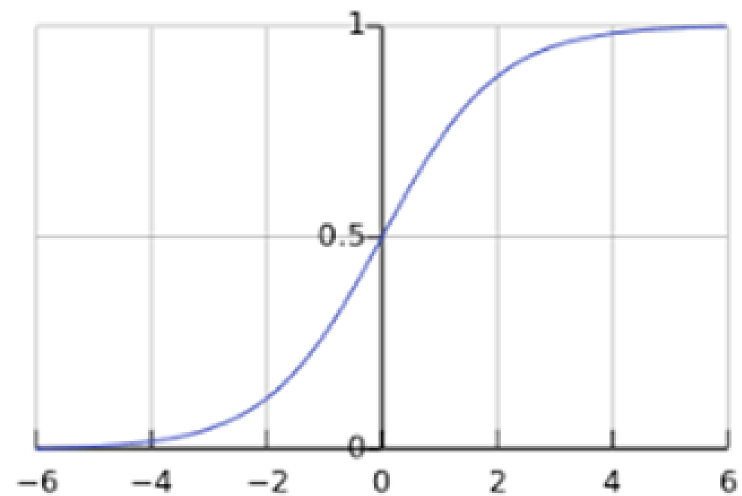
$$\begin{matrix} x_1 \\ x_2 \end{matrix} \xrightarrow{W} \Sigma \xrightarrow{z} \sigma(z) \longrightarrow y_1$$

- $W = \begin{bmatrix} -7 \\ -7 \end{bmatrix}, b = 3$

$$\begin{matrix} x_1 \\ x_2 \end{matrix} \xrightarrow{W} \Sigma \xrightarrow{z} \sigma(z) \longrightarrow y_2$$

- $W = \begin{bmatrix} -11 \\ -11 \end{bmatrix}, b = 6$

$$\begin{matrix} x_1 \\ x_2 \end{matrix} \xrightarrow{W} \Sigma \xrightarrow{z} \sigma(z) \longrightarrow y_3$$



퍼셉트론 예

$W = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, b = -8$ $\quad$ $W = \begin{bmatrix} -7 \\ -7 \end{bmatrix}, b = 3$ $\quad$ $W = \begin{bmatrix} -11 \\ -11 \end{bmatrix}, b = 6$

$case \quad x_1 = 0, \qquad x_2 = 0$

$[0 \quad 0]\begin{bmatrix} 5 \\ 5 \end{bmatrix} - 8 = 0 + 0 - 8, \quad y_1 = \mathrm{s}(-8) = 0$

$[0 \quad 0]\begin{bmatrix} -7 \\ -7 \end{bmatrix} + 3 = 0 + 0 + 3, \quad y_2 = \mathrm{s}(3) = 1$

$[0 \quad 1]\begin{bmatrix} -11 \\ -11 \end{bmatrix} + 6 = 0 - 11 + 6, \; \hat{y} = \mathrm{s}(-5) = 0$

$x_1 \quad x_2$	$y_1 \quad y_2$	$\hat{y}$	XOR
0 0	0 1	0	0
0 1			1
1 0			1
1 1			0

퍼셉트론 예

$$W = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, b = -8 \qquad W = \begin{bmatrix} -7 \\ -7 \end{bmatrix}, b = 3 \qquad W = \begin{bmatrix} -11 \\ -11 \end{bmatrix}, b = 6$$

$$case \quad x_1 = 0, \qquad x_2 = 1$$

$$[0 \quad 1]\begin{bmatrix} 5 \\ 5 \end{bmatrix} - 8 = 0 + 5 - 8, \quad y_1 = s(-3) = 0$$

$$[0 \quad 1]\begin{bmatrix} -7 \\ -7 \end{bmatrix} + 3 = 0 - 7 + 3, \quad y_2 = s(-4) = 0$$

$$[0 \quad 0]\begin{bmatrix} -11 \\ -11 \end{bmatrix} + 6 = 0 + 0 + 6, \quad \hat{y} = s(6) = 1$$

$x_1 \quad x_2$	$y_1 \quad y_2$	$\hat{y}$	XOR
0 0	0 1	0	0
0 1	0 0	1	1
1 0			1
1 1			0

퍼셉트론 예

$W = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, b = -8$ $\quad W = \begin{bmatrix} -7 \\ -7 \end{bmatrix}, b = 3$ $\quad W = \begin{bmatrix} -11 \\ -11 \end{bmatrix}, b = 6$

$case \quad x_1 = 1, \quad x_2 = 0$

$$[1 \quad 0] \begin{bmatrix} 5 \\ 5 \end{bmatrix} - 8 = 5 + 0 - 8, \quad y_1 = s(-3) = 0$$

$$[1 \quad 0] \begin{bmatrix} -7 \\ -7 \end{bmatrix} + 3 = -7 + 0 + 3, \quad y_2 = s(-4) = 0$$

$$[0 \quad 0] \begin{bmatrix} -11 \\ -11 \end{bmatrix} + 6 = 0 + 0 + 6, \quad \hat{y} = s(6) = 1$$

$x_1 \quad x_2$	$y_1 \quad y_2$	$\hat{y}$	XOR
0 0	0 1	0	0
0 1	0 0	1	1
1 0	0 0	1	1
1 1			0

퍼셉트론 예

$$W = \begin{bmatrix}5\\5\end{bmatrix}, b = -8 \qquad W = \begin{bmatrix}-7\\-7\end{bmatrix}, b = 3 \qquad W = \begin{bmatrix}-11\\-11\end{bmatrix}, b = 6$$

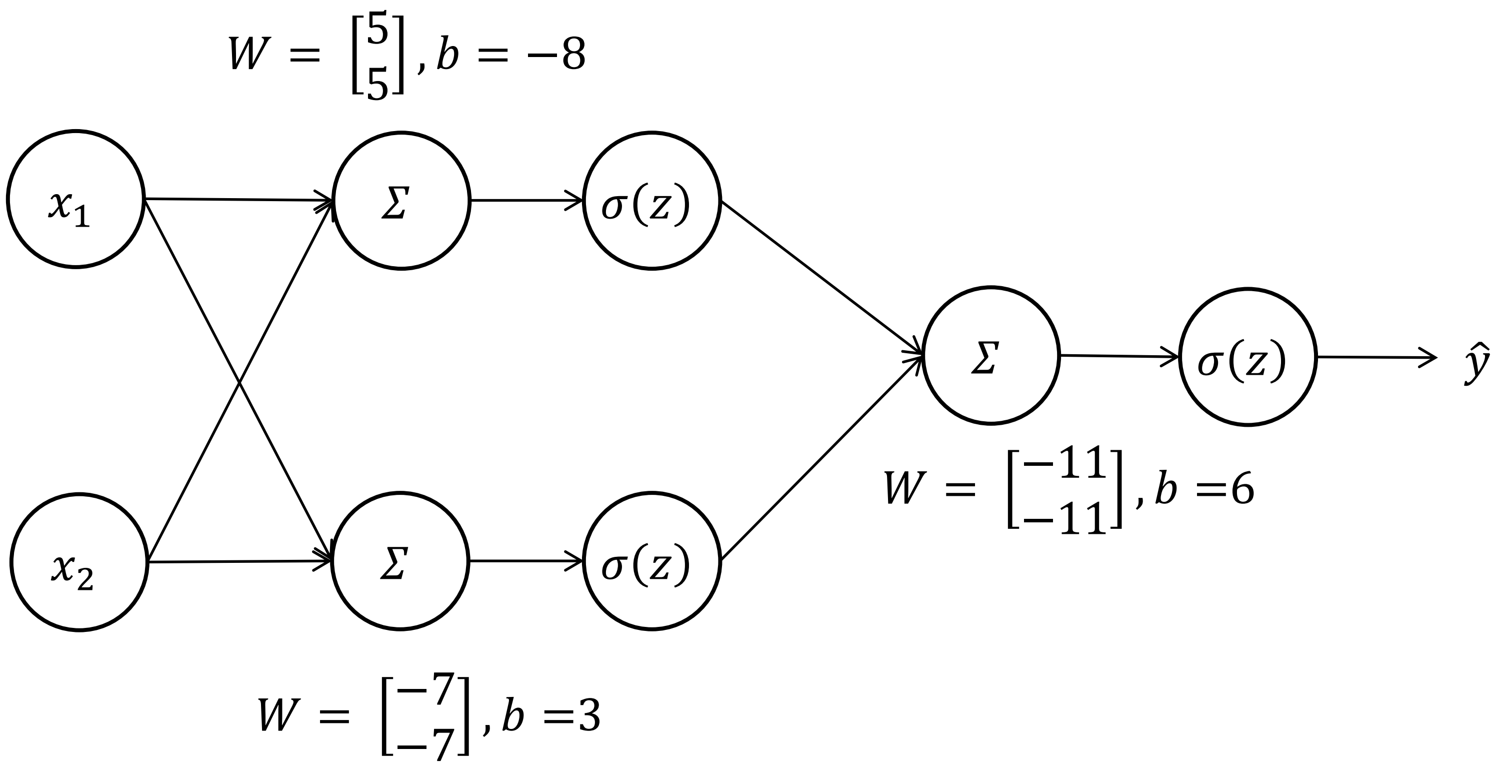
$case \quad x_1 = 1, \qquad x_2 = 1$

$$[1 \quad 1]\begin{bmatrix}5\\5\end{bmatrix} - 8 = 5 + 5 - 8, \quad y_1 = s(2) = 1$$

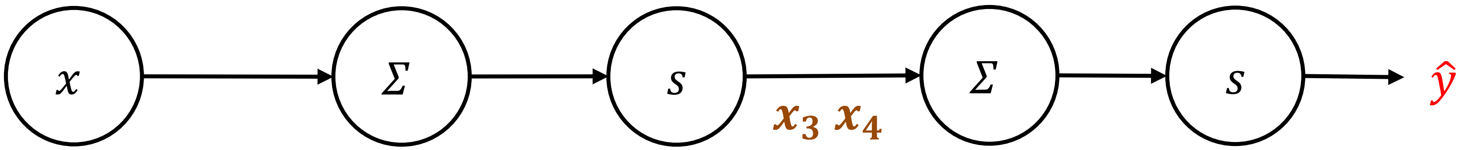
$$[1 \quad 1]\begin{bmatrix}-7\\-7\end{bmatrix} + 3 = -7 - 7 + 3, \; y_2 = s(-11) = 0$$

$$[1 \quad 0]\begin{bmatrix}-11\\-11\end{bmatrix} + 6 = -11 + 0 + 6, \quad \hat{y} = s(-5) = 0$$

$x_1 \quad x_2$	$y_1 \quad y_2$	$\hat{y}$	XOR
0 0	0 1	0	0
0 1	0 0	1	1
1 0	0 0	1	1
1 1	1 0	0	0


$W = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, b = -8$
x_1
x_2
Σ
Σ
$\sigma(z)$
$\sigma(z)$
Σ
$\sigma(z)$
$\hat{y}$
$W = \begin{bmatrix} -11 \\ -11 \end{bmatrix}, b = 6$
$W = \begin{bmatrix} -7 \\ -7 \end{bmatrix}, b = 3$

$$W_1 = \begin{bmatrix} 5 & -7 \\ 5 & -7 \end{bmatrix}, b_1 = [-8 \quad 3]$$



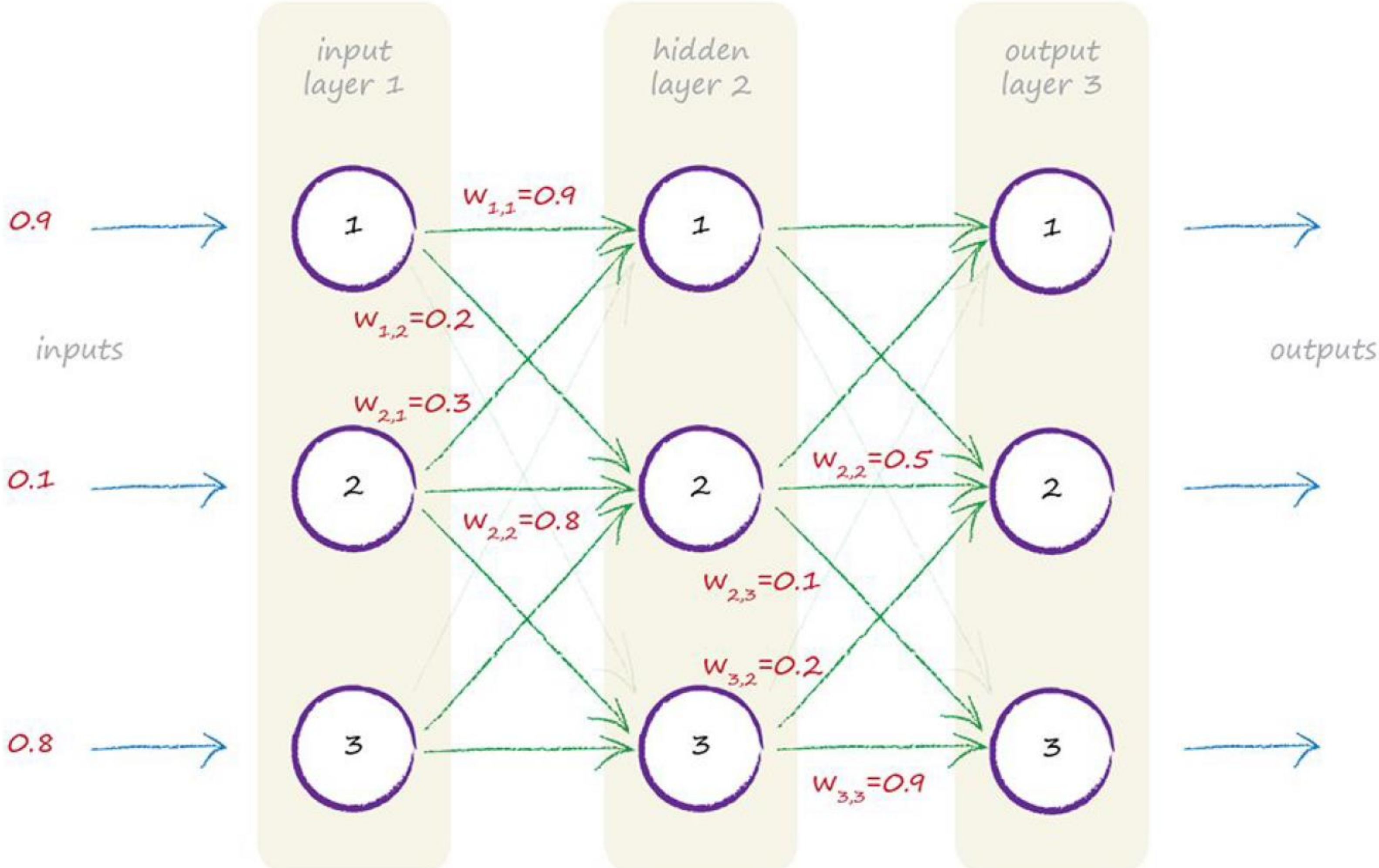
$[0 \quad 0]$
$[0 \quad 1]$
$[1 \quad 0]$
$[1 \quad 1]$

$$W_2 = \begin{bmatrix} -11 \\ -11 \end{bmatrix}, b_2 = 6$$

$$[x_1 \;\; x_2]\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} + [b_{11} \;\; b_{12}]$$
$$= sigmoid([x_1 w_{11} + x_2 w_{21} \quad x_1 w_{12} + x_2 w_{22}] + [b_{11} \;\; b_{12}])$$
$$= [x_3 \;\; x_4]$$

$$[x_3 \;\; x_4]\begin{bmatrix} w_{21} \\ w_{23} \end{bmatrix} + [b_2]$$
$$= sigmoid([x_3 w_{21} + x_4 w_{23}] + [b_2]) = [\hat{y}]$$

3계층 신경망에 행렬곱 적용



ref : Make your own neural network - Triq Rashid

입력값

$$I = \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix}$$

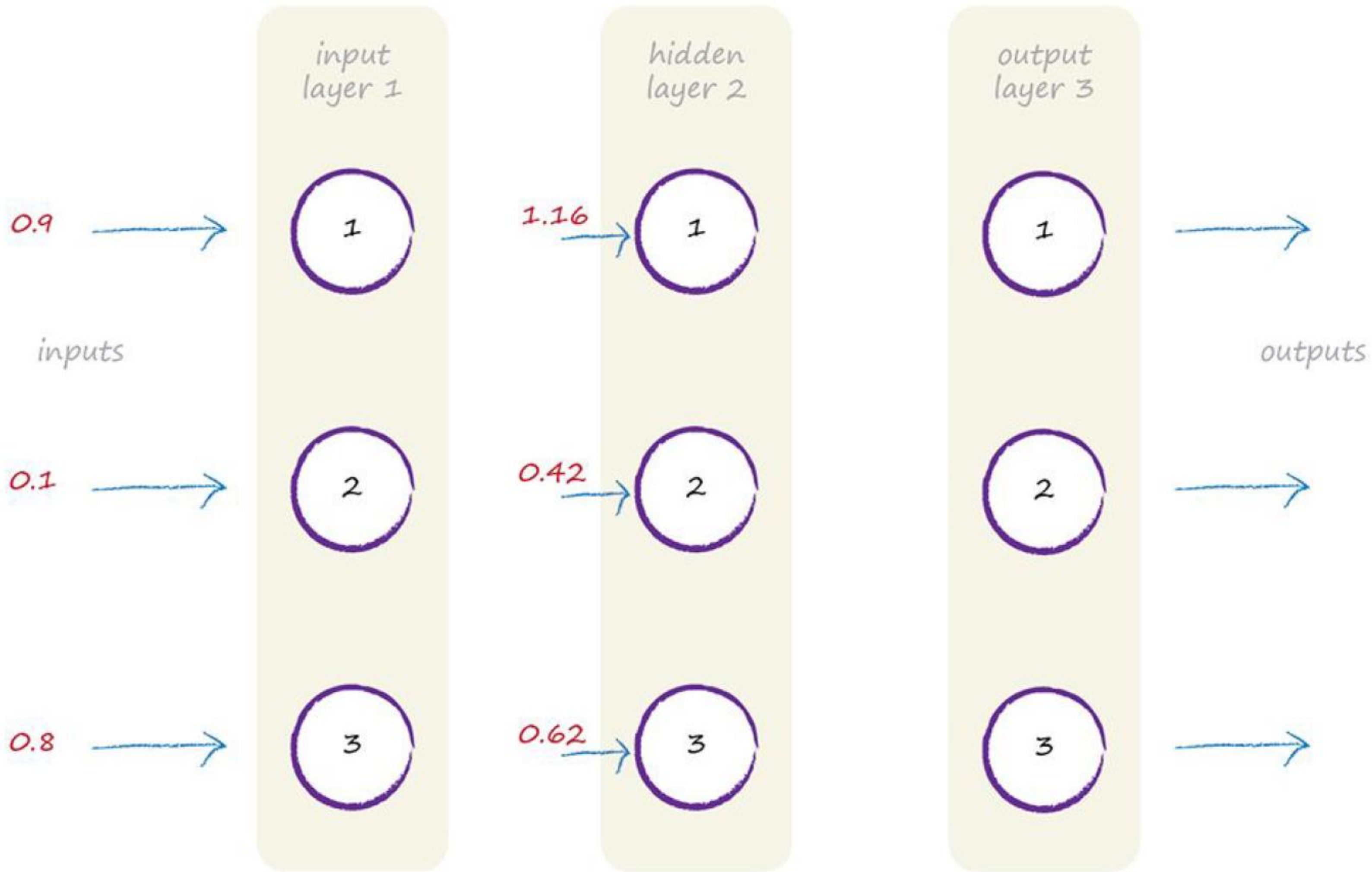
$$W_{i\_hidden} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix}$$

$$W_{o\_hidden} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix}$$

$$X_{hidden} = \begin{pmatrix} 0.9 & 0.3 & 0.4 \\ 0.2 & 0.8 & 0.2 \\ 0.1 & 0.5 & 0.6 \end{pmatrix} \begin{pmatrix} 0.9 \\ 0.1 \\ 0.8 \end{pmatrix} = \begin{pmatrix} 1.16 \\ 0.42 \\ 0.62 \end{pmatrix}$$

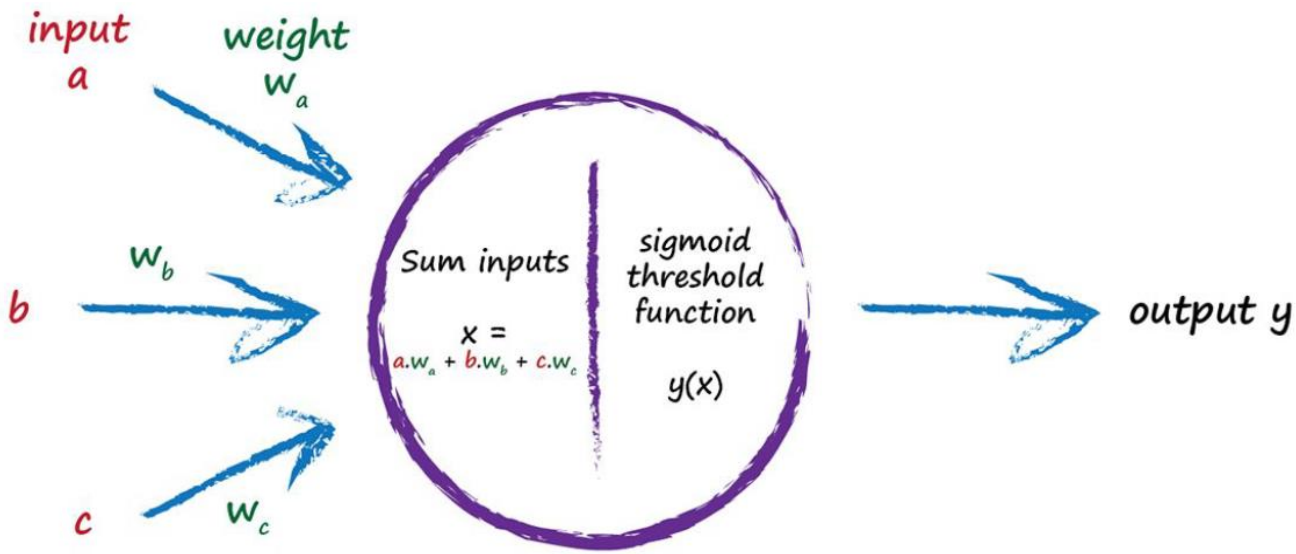
ref : Make your own neural network - Triq Rashid

Hidden Layer 입력값



ref : Make your own neural network - Triq Rashid

sigmoid function



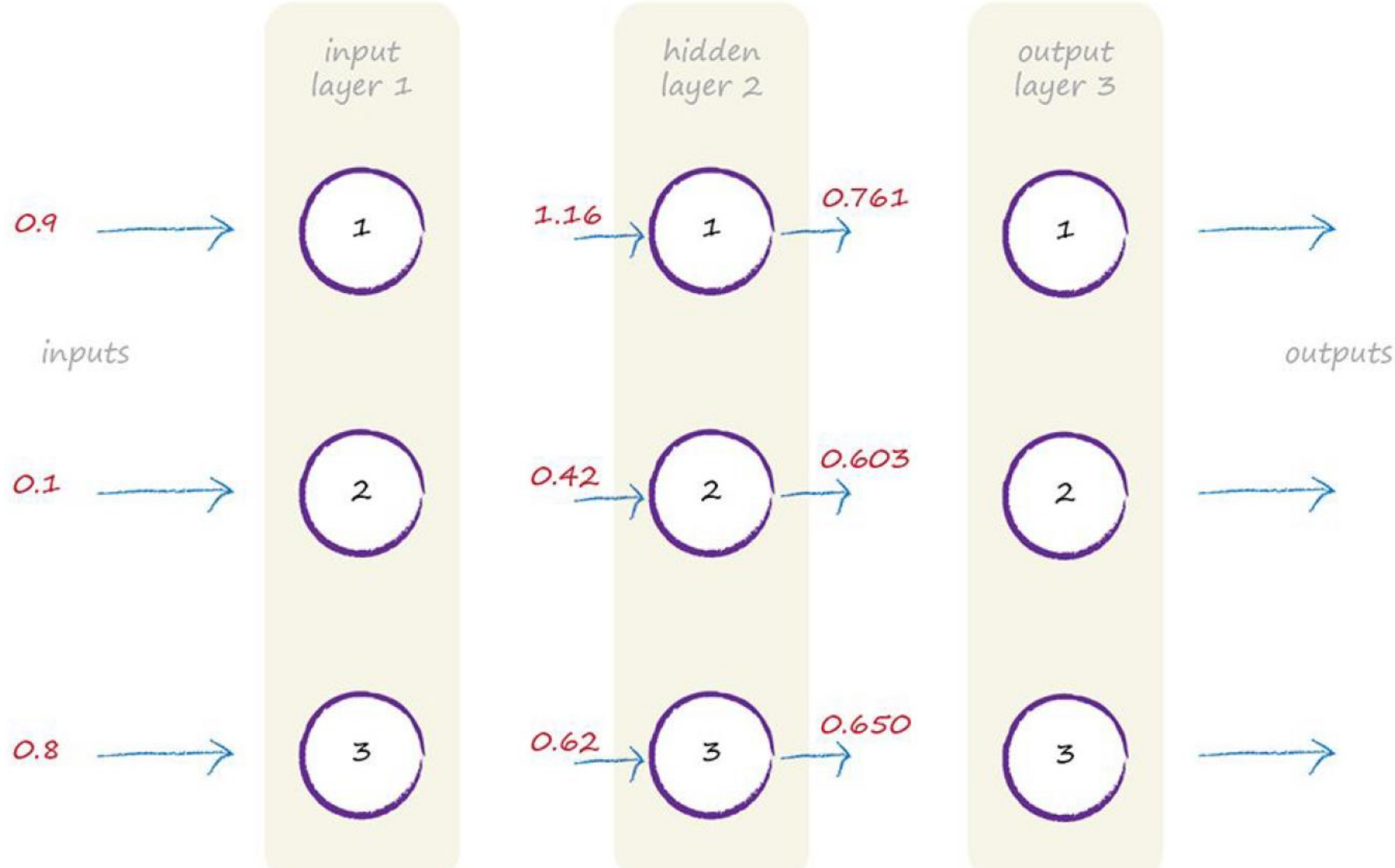
$$O_{hidden} = sigmoid(X_{hidden}) = \text{sigmoid}\begin{pmatrix}1.16\\0.42\\0.62\end{pmatrix} = \begin{pmatrix}0.761\\0.603\\0.650\end{pmatrix}$$

$$y = \frac{1}{1+e^{-x}}$$, ($x = 1.16$ 대입하면 $e^{-x} = 0.3135$, $e = 2.71828 \ldots\ldots$)

$$= \frac{1}{1+0.3135} = \mathbf{0.761}$$

ref : Make your own neural network - Triq Rashid

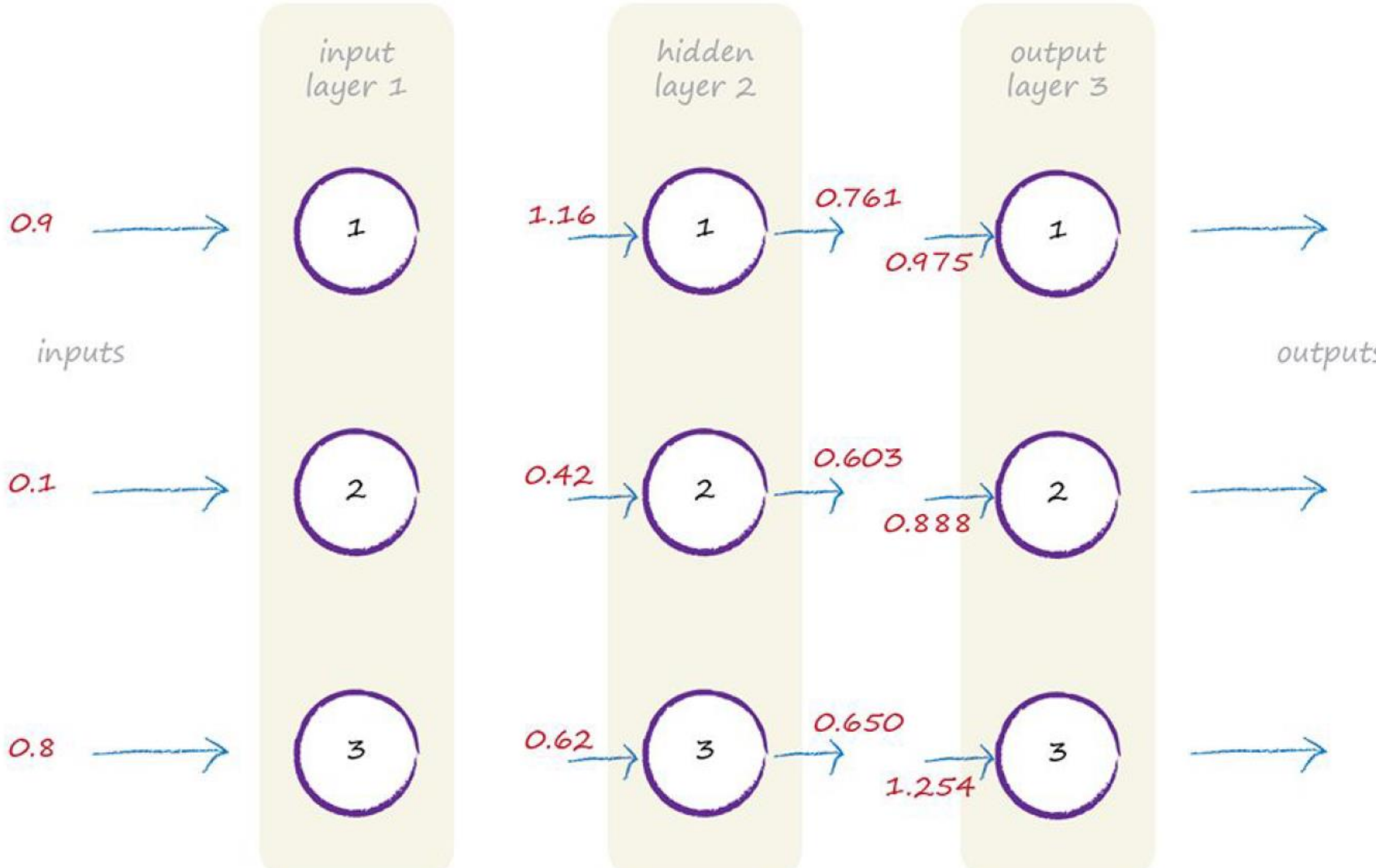
Hidden Layer결과값



ref : Make your own neural network - Triq Rashid

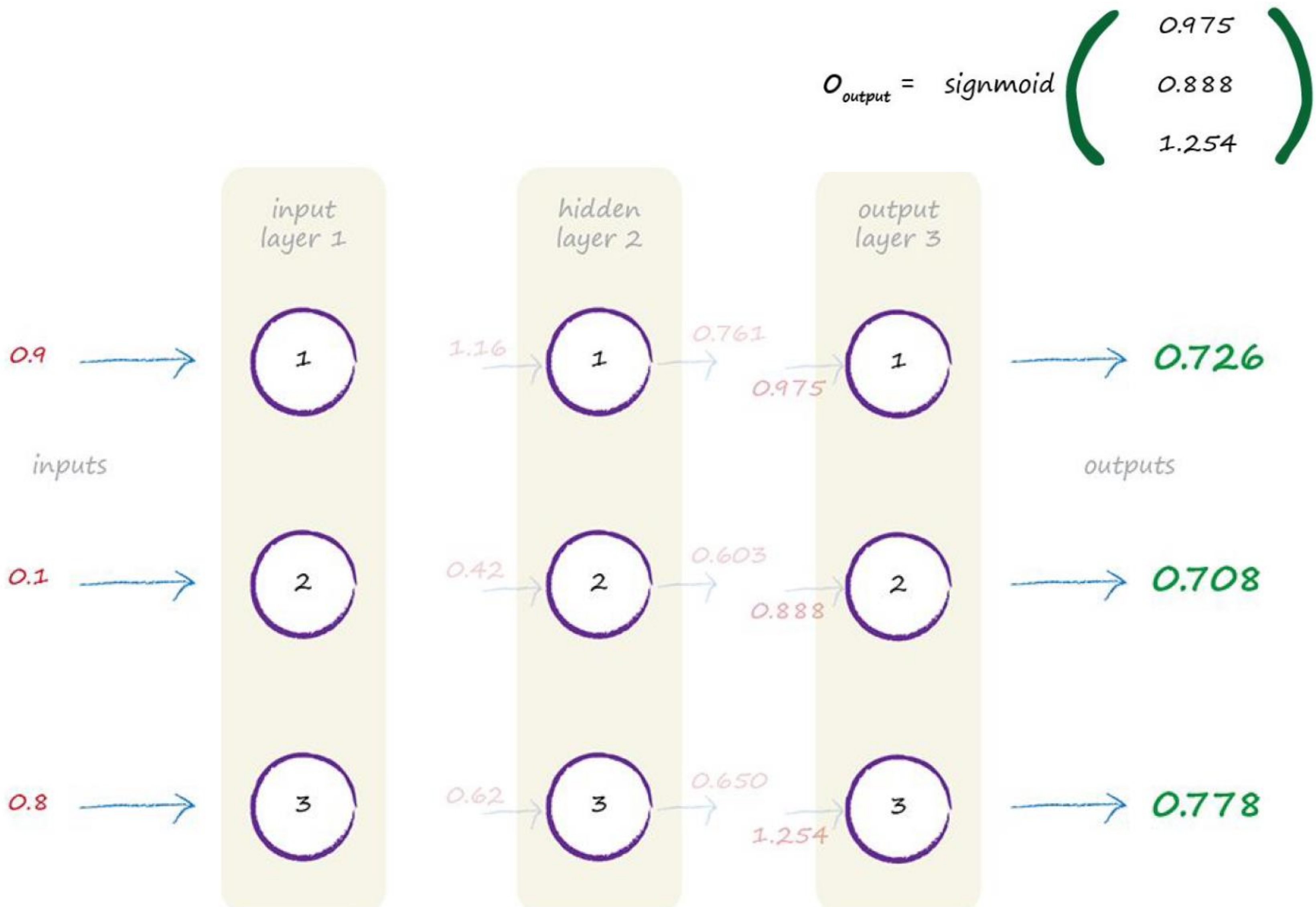
Ouput Layer입력

$$X_{output} = \begin{pmatrix} 0.3 & 0.7 & 0.5 \\ 0.6 & 0.5 & 0.2 \\ 0.8 & 0.1 & 0.9 \end{pmatrix} \cdot \begin{pmatrix} 0.761 \\ 0.603 \\ 0.650 \end{pmatrix} = \begin{pmatrix} 0.975 \\ 0.888 \\ 1.254 \end{pmatrix}$$



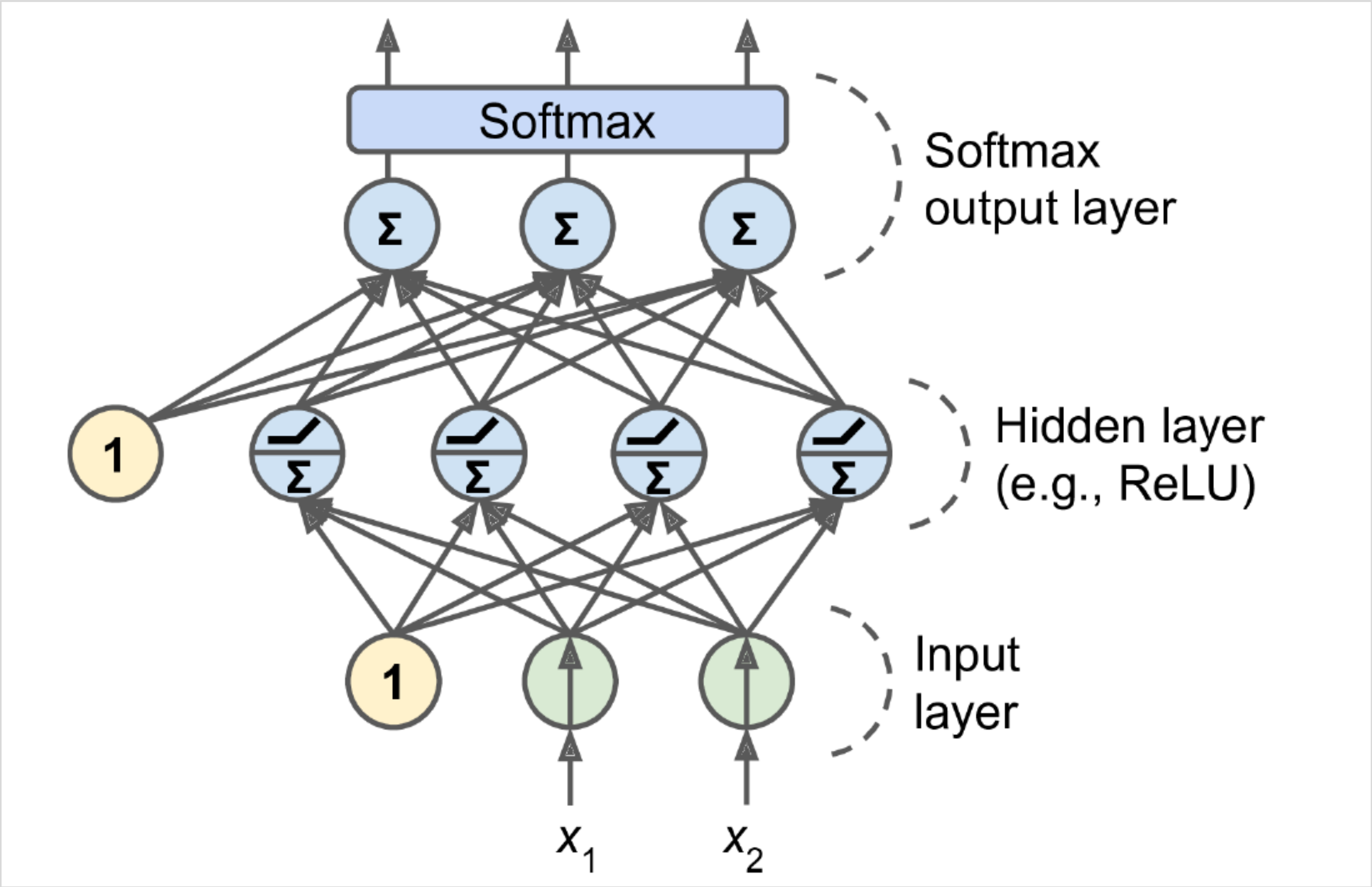
ref : Make your own neural network - Triq Rashid

Output Layer결과값(최종결과값)



ref : Make your own neural network - Triq Rashid

분류를 위한 다층 퍼셉트론



텐서플로 케라스

```
import tensorflow as tf
from tensorflow import keras

tf.__version__   # 2.3.0
keras.__version__   # 2.4.0

# 패션 MNIST 데이터셋 로드
fashion_mnist = keras.datasets.fashion_mnist
(X_train_full, y_train_full), (X_test, y_test) = fashion_mnist.load_data()

# 훈련 세트는 60,000개. 각 이미지의 크기는 28x28 픽셀
X_train_full.shape   # (60000, 28, 28)
X_train_full.dtype   # dtype('uint8')

# 전체 훈련 세트를 검증 세트와 훈련 세트로 나눈다.
X_valid, X_train = X_train_full[:5000] / 255., X_train_full[5000:] / 255.
y_valid, y_train = y_train_full[:5000], y_train_full[5000:]
X_test = X_test /  255.0   # 픽셀 강도를 255로 나누어 0~1 범위의 실수로 변경

plt.imshow(X_train[0], cmap="binary")
plt.axis('off')
plt.show()
```

```
y_train      # [4, 0, 7, ..., 3, 0, 5]
class_names = ["T-
shirt/top", "Trouser", "Pullover", "Dress", "Coat",
               "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot"]
class_names[y_train[0]]   # Coat
# 검증 세트는 5,000개의 이미지
X_valid.shape   # (5000, 28, 28)
# 테스트 세트는 10,000개의 이미지
X_test.shape   #  (10000, 28, 28)
```

```
n_rows = 4
n_cols = 10
plt.figure(figsize=(n_cols * 1.2, n_rows * 1.2))
for row in range(n_rows):
    for col in range(n_cols):
        index = n_cols * row + col
        plt.subplot(n_rows, n_cols, index + 1)
        plt.imshow(X_train[index], cmap="binary", interpolation="nearest")
        plt.axis('off')
        plt.title(class_names[y_train[index]], fontsize=12)
plt.subplots_adjust(wspace=0.2, hspace=0.5)
plt.show()
```

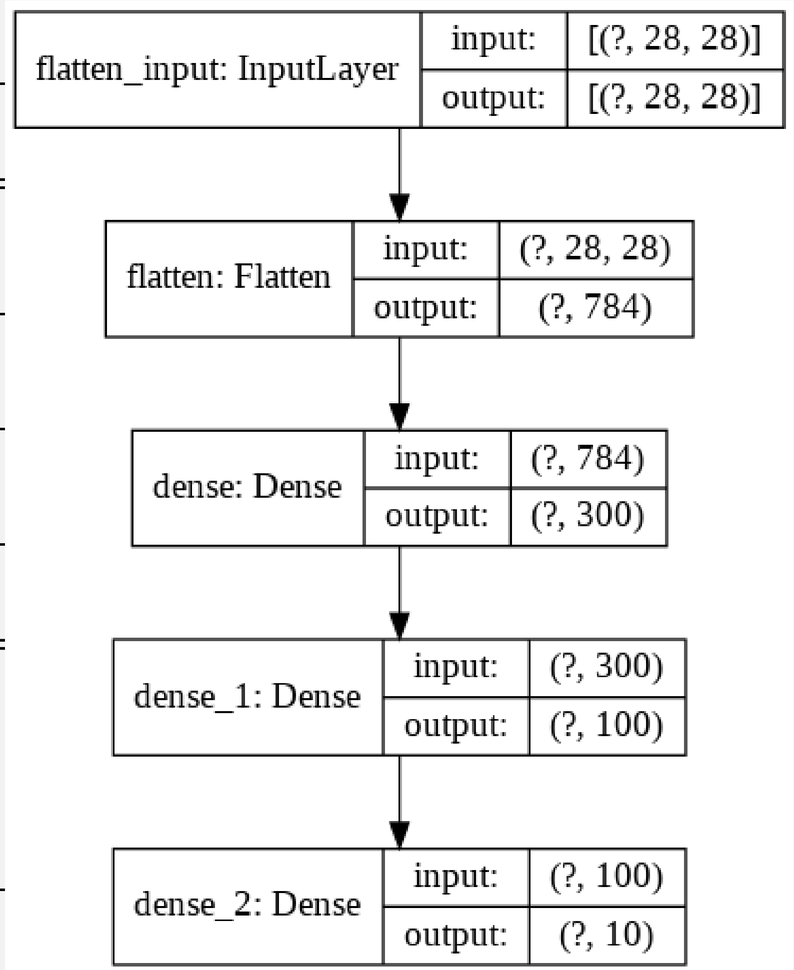
Sequential()을 사용하여 MLP를 구축, 훈련, 평가, 예측하는 방법


```
# 두 개의 은닉층으로 이루어진 분류용 다층 퍼셉트론
model = keras.models.Sequential()
model.add(keras.layers.Flatten(input_shape=[28, 28]))
model.add(keras.layers.Dense(300, activation="relu"))
model.add(keras.layers.Dense(100, activation="relu"))
model.add(keras.layers.Dense(10, activation="softmax"))

# 방법 2 - Sequential 모델에 층의 리스트를 전달
model = keras.models.Sequential([
    keras.layers.Flatten(input_shape=[28, 28]),
    keras.layers.Dense(300, activation="relu"),
    keras.layers.Dense(100, activation="relu"),
    keras.layers.Dense(10, activation="softmax")
])
```

summary()

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 784)               0
_________________________________________________________________
dense (Dense)                (None, 300)               235500
_________________________________________________________________
dense_1 (Dense)              (None, 100)               30100
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1010
=================================================================
Total params: 266,610
Trainable params: 266,610
Non-trainable params: 0
_________________________________________________________________
```



```
model.layers
```

```
[<tensorflow.python.keras.layers.core.Flatten at 0x7f19fd9e7518>,
 <tensorflow.python.keras.layers.core.Dense at 0x7f19fd9e76a0>,
 <tensorflow.python.keras.layers.core.Dense at 0x7f19fd9e7908>,
 <tensorflow.python.keras.layers.core.Dense at 0x7f19fd9e7ba8>]
```

```
hidden1 = model.layers[1]
hidden1.name    # dense
model.get_layer(hidden1.name) is hidden1   # True

# 층의 모든 파라미터는 get_wights()함수에서
weights, biases = hidden1.get_weights()
weights
```

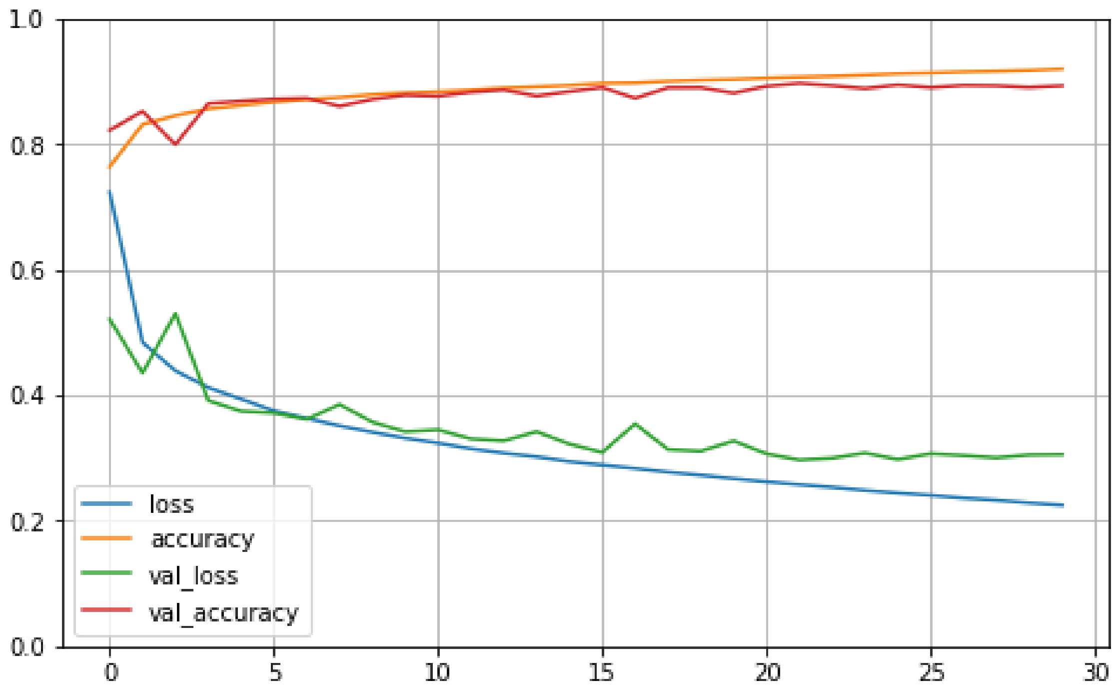
```
array([[ 0.02448617, -0.00877795, -0.02189048, ..., -0.02766046,
         0.03859074, -0.06889391],
       [ 0.00476504, -0.03105379, -0.0586676 , ...,  0.00602964,
        -0.02763776, -0.04165364],
       [-0.06189284, -0.06901957,  0.07102345, ..., -0.04238207,
         0.07121518, -0.07331658],
       ...,
       [-0.03048757,  0.02155137, -0.05400612, ..., -0.00113463,
         0.00228987,  0.05581069],
       [ 0.07061854, -0.06960931,  0.07038955, ..., -0.00384101,
         0.00034875,  0.02878492],
       [-0.06022581,  0.01577859, -0.02585464, ..., -0.00527829,
         0.00272203, -0.06793761]], dtype=float32)
```

모델 컴파일


```
# 손실함수와 최적화 지정
model.compile(loss="sparse_categorical_crossentropy",
              optimizer="sgd",
              metrics=["accuracy"])

# 모델 훈련 : epochs = 30
history = model.fit(X_train, y_train, epochs=30,
                    validation_data=(X_valid, y_valid))

# fit()메서드가 반환하는 훈련 파라미터
history.params

# 수행된epoch 리스트
print(history.epoch)

# 훈련세트와 검증 세트에 대한 손실과 측정한 지표를 담은 딕셔너리
history.history.keys()
```

학습곡선


```
# epoch마다 측정한 평균적인 훈련 손실과 정확도 및 epoch종료시점마다 측정한 평균적인 검증 손실과 정확도
import pandas as pd

pd.DataFrame(history.history).plot(figsize=(8, 5))
plt.grid(True)
plt.gca().set_ylim(0, 1)
plt.show()
```



```
model.evaluate(X_test, y_test)
```

313/313 [==============================] - 0s 1ms/step - **loss: 0.3382 - accuracy: 0.8822**
[0.3381877839565277, 0.8822000026702881]

```
# 예측 : 테스트 세트의 3개 샘플 사용
X_new = X_test[:3]
y_proba = model.predict(X_new)
y_proba.round(2)
```

array([[0. , 0. , 0. , 0. , 0. , 0.01, 0. , 0.03, 0. , **0.96**],
[0. , 0. , **0.99**, 0. , 0.01, 0. , 0. , 0. , 0. , 0.],
[0. , 1. , 0. , 0. , 0. , 0. , 0. , 0. , 0. , 0.]],
dtype=float32)

```
y_pred = model.predict_classes(X_new)
y_pred  # array([9, 2, 1])
np.array(class_names)[y_pred]
```

array(['Ankle boot', 'Pullover', 'Trouser'], dtype='<U11')

```
y_new = y_test[:3]
y_new    # array([9, 2, 1], dtype=uint8)
```

```python
plt.figure(figsize=(7.2, 2.4))
for index, image in enumerate(X_new):
    plt.subplot(1, 3, index + 1)
    plt.imshow(image, cmap="binary", interpolation="nearest")
    plt.axis('off')
    plt.title(class_names[y_test[index]], fontsize=12)
plt.subplots_adjust(wspace=0.2, hspace=0.5)
plt.show()
```