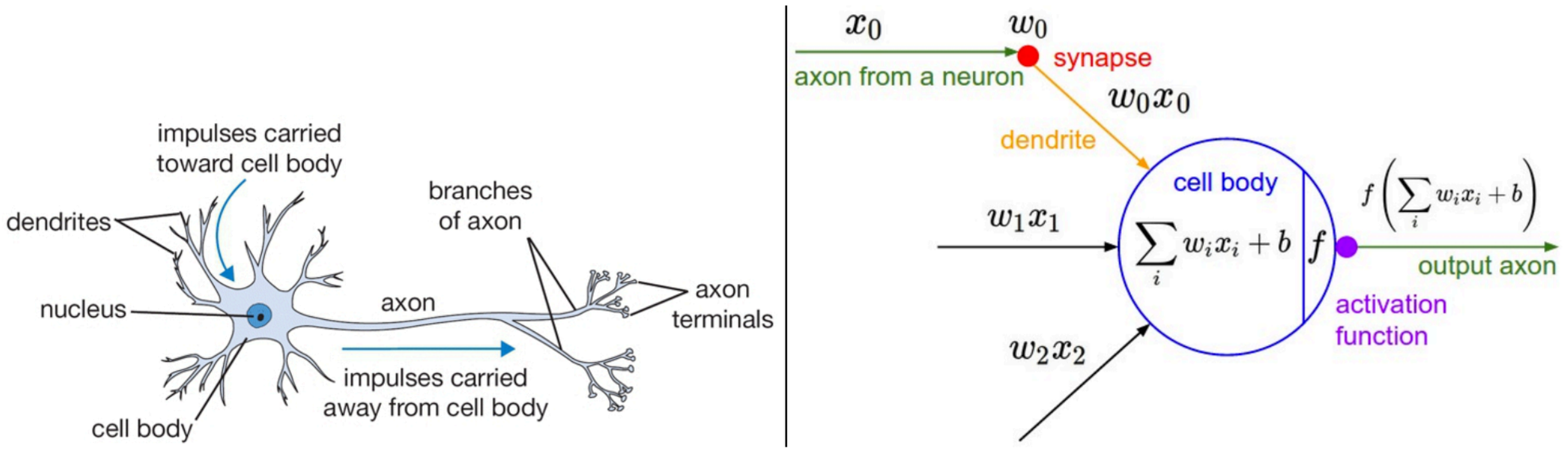
블록과 함께하는 파이썬
딥러닝 케라스

Part3 1~2

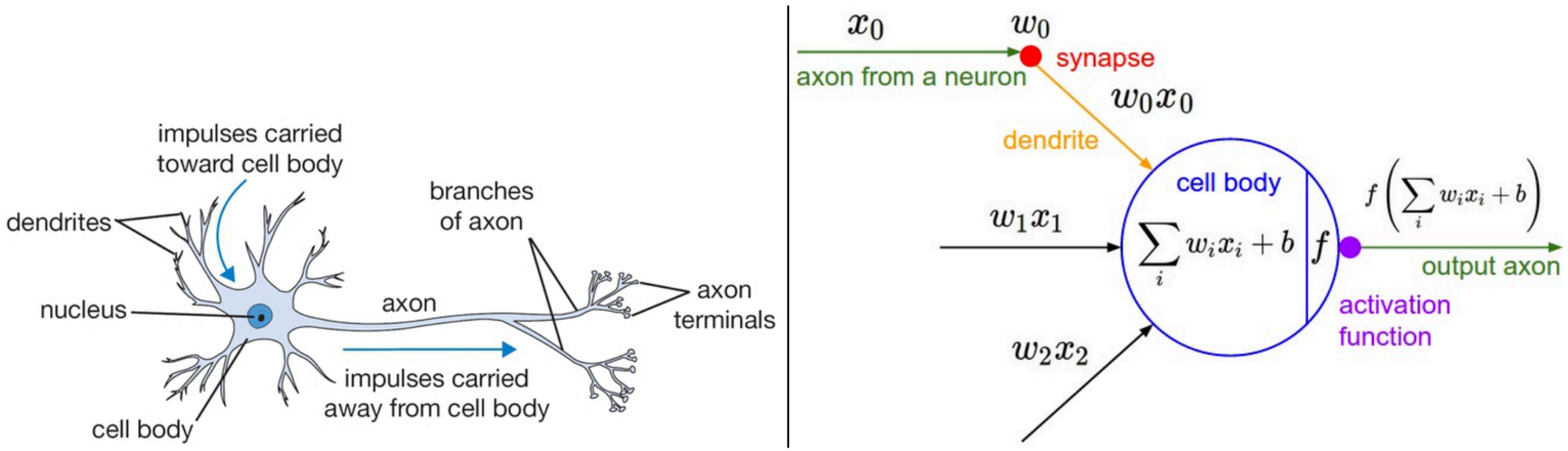
다층 퍼셉트론 레이어 이야기



A cartoon drawing of a biological neuron (left) and its mathematical model (right).

- **axon (축삭돌기) : 팔처럼 몸체에서 뻗어나와 다른 뉴런의 수상돌기와 연결됩니다.**
- **dendrite (수상돌기) : 다른 뉴런의 축삭 돌기와 연결되며, 몸체에 나뭇가지 형태로 붙어 있습니다.**
- **synapse (시냅스) : 축사돌기와 수상돌기가 연결된 지점입니다. 여기서 한 뉴런이 다른 뉴런으로 신호가 전달됩니다.**

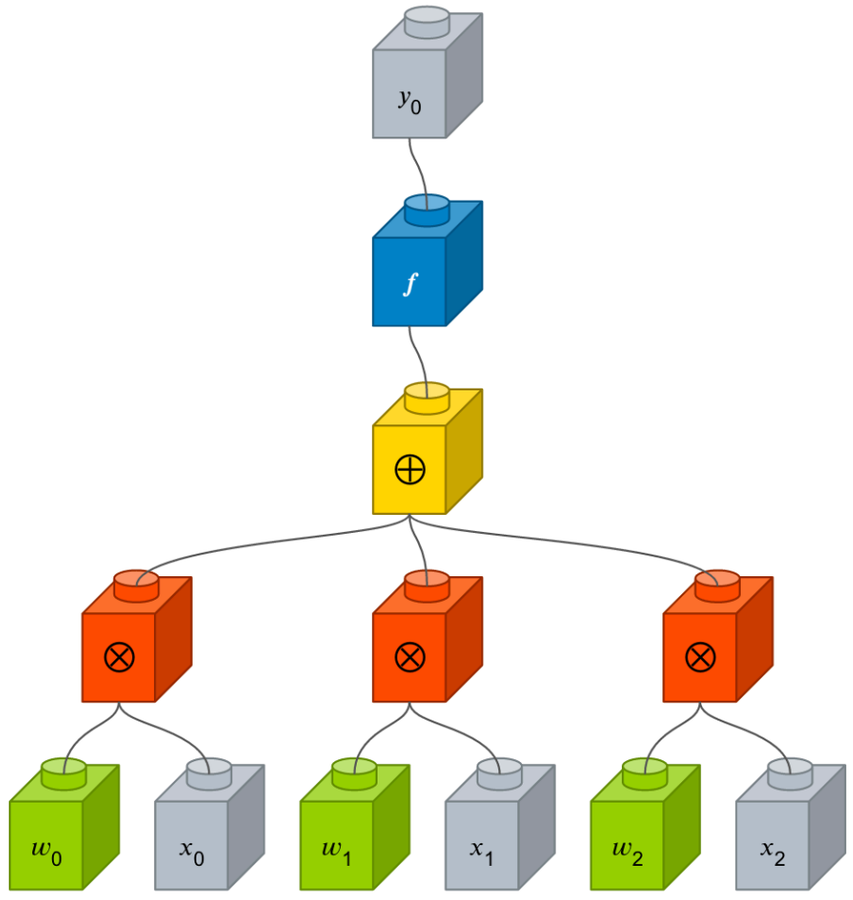
다층 퍼셉트론 레이어 이야기



A cartoon drawing of a biological neuron (left) and its mathematical model (right).

- **x0, x1, x2 : 입력되는 뉴런의 축삭돌기로부터 전달되는 신호의 양**
- **w0, w1, w2 : 시냅스의 강도, 즉 입력되는 뉴런의 영향력을 나타냅니다.**
- **w0x0 + w1x1 + w2*x2 : 입력되는 신호의 양과 해당 신호의 시냅스 강도가 곱해진 값의 합계**
- **f : 최종 합계가 다른 뉴런에게 전달되는 신호의 양을 결정짓는 규칙, 이를 활성화 함수라고 부릅니다.**

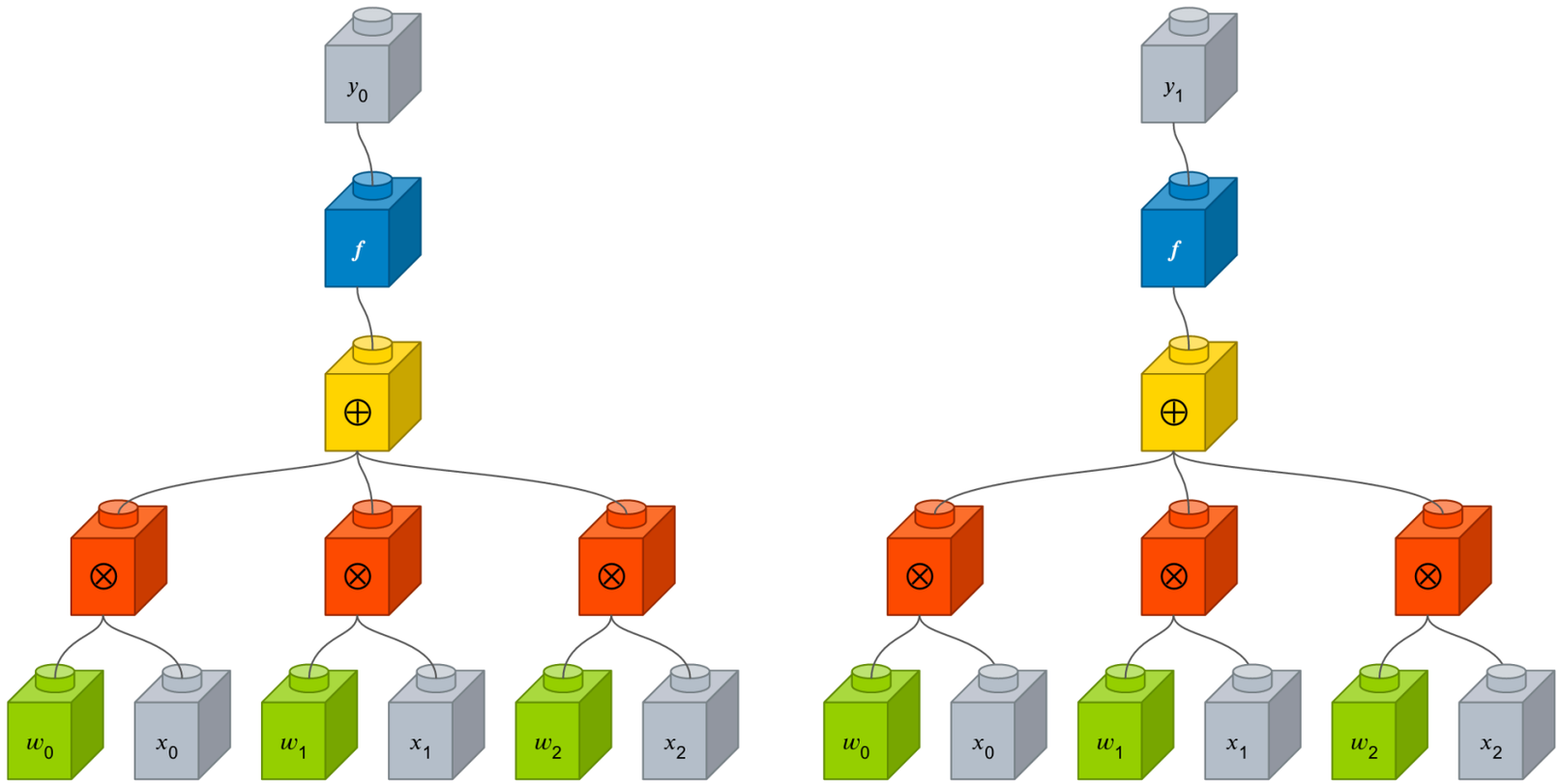
다층 퍼셉트론 레이어 이야기



,,

- 녹색 블럭은 시냅스의 강도 w
- 회색 블럭은 입력 값 x
- 노란색과 빨간색 블럭은 연산자
- 파란색 블럭은 활성화 함수 f

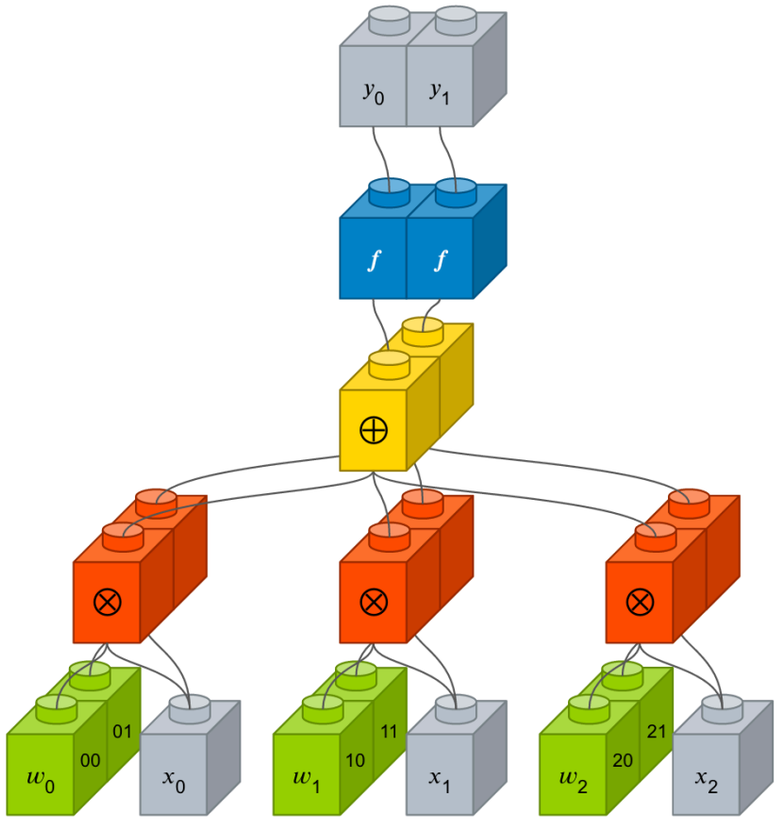
다층 퍼셉트론 레이어 이야기



,,

- 만약 세 개의 신호가 서로다른 뉴런 2개에 입력된다면 ? 출력은 총 2개
- 입력은?

다층 퍼셉트론 레이어 이야기



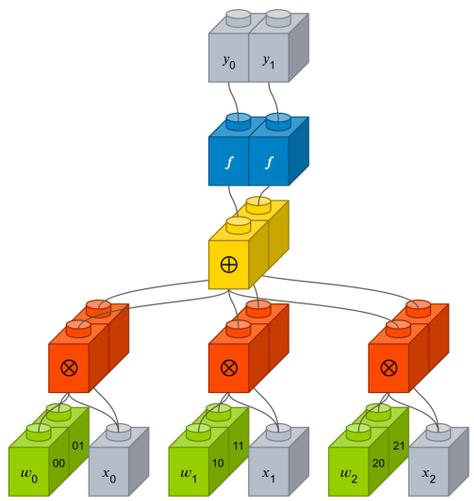
,,

- 3개의 신호를 받는 2개의 뉴런 연결의 수는 3*2 = 6 == 시냅스의 강도
  녹색 블럭

Layer

- Dense
- Convolutional
- Fully connected
- Pooling layer
- Normalisation

입출력을 모두 연결해주는 Dense 레이어



- 입력 수 * 출력 수 = 연결 수
- 연결 수는 가중치(연결 강도)이다.
- 가중치는 높을 수록 해당 뉴런에 영향을 크다
- Ex) 성별을 판단하는 문제에서 머리카락, 키, 혈액형이 있다면,
  가중치는 머리카락 , 키 > 혈액형일 것이다.
  딥러닝 학습과정에서는 이러한 가중치들이 조정된다.
- 이렇게 출력과 입력을 모두 연결하는 레이어를 Dense(전결합층)이라고 한다.

입출력을 모두 연결해주는 Dense 레이어

```
Dense(8, input_dim=4, init='uniform', activation='relu'))
```

- 첫번째 인자 : 출력 뉴런의 수를 설정합니다.
- input_dim : 입력 뉴런의 수를 설정합니다.
- init : 가중치 초기화 방법 설정합니다.
  - ‘uniform’ : 균일 분포
  - ‘normal’ : 가우시안 분포
- activation : 활성화 함수 설정합니다.
  - ‘linear’ : 디폴트 값, 입력뉴런과 가중치로 계산된 결과값이 그대로 출력으로 나옵니다.
  - ‘relu’ : rectifier 함수, 은익층에 주로 쓰입니다.
  - ‘sigmoid’ : 시그모이드 함수, 이진 분류 문제에서 출력층에 주로 쓰입니다.
  - ‘softmax’ : 소프트맥스 함수, 다중 클래스 분류 문제에서 출력층에 주로 쓰입니다.

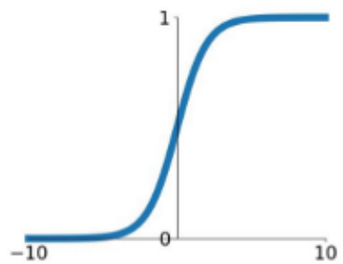
*** Dense Layer 는 입력 뉴련수에 상관 없이 출력 뉴런 수를 설정 가능**

Activation Function

Activation Functions

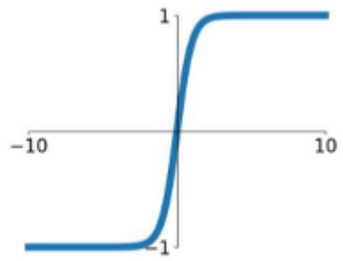
Sigmoid

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



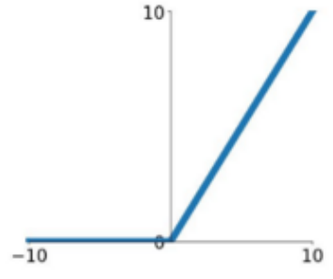
tanh

$$\tanh(x)$$



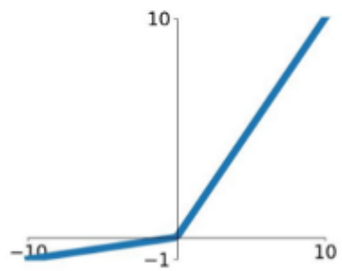
ReLU

$$\max(0, x)$$



Leaky ReLU

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

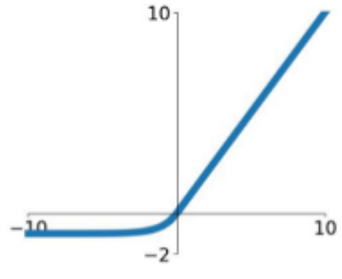


Maxout

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

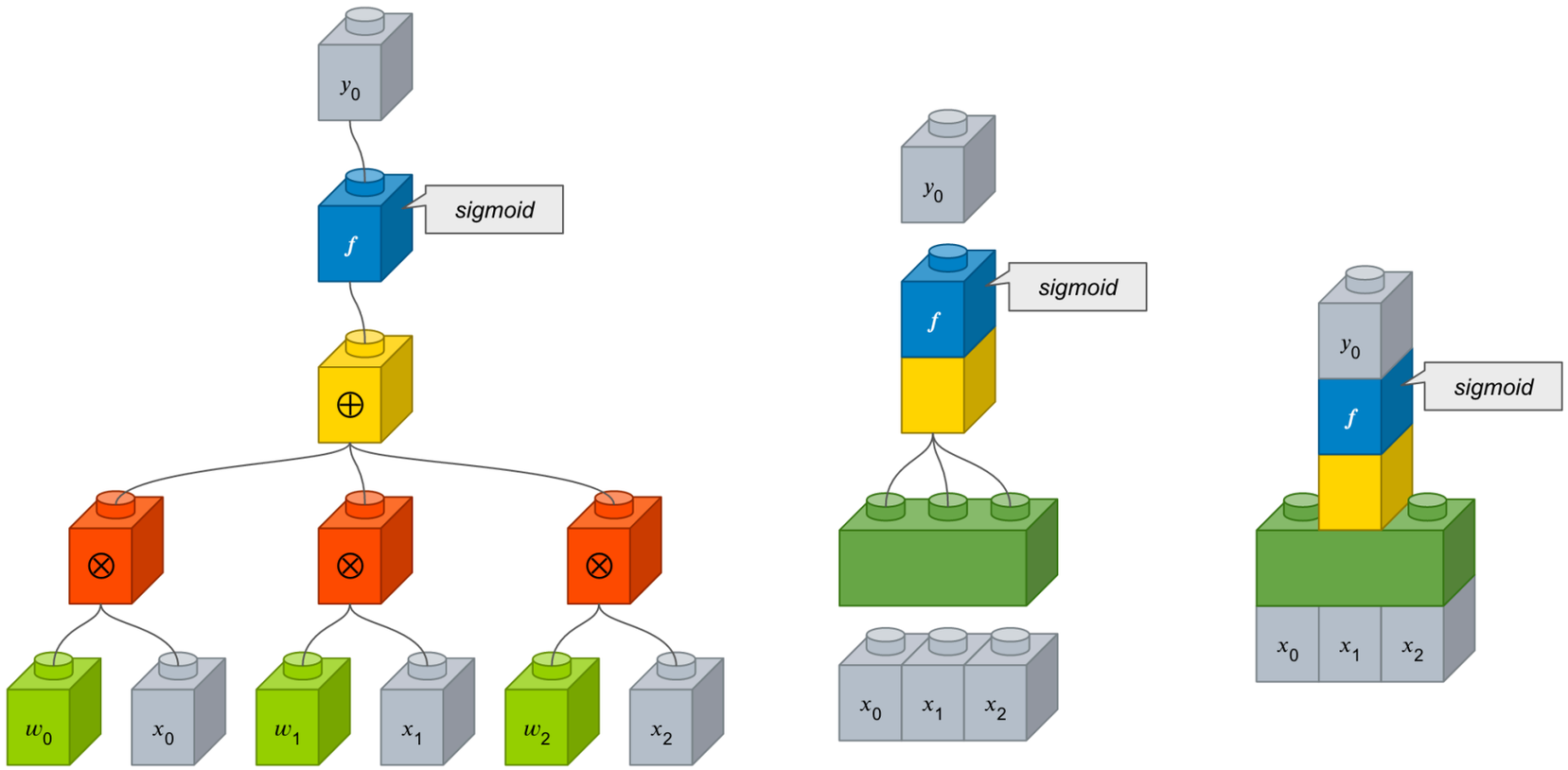
ELU

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



입출력을 모두 연결해주는 Dense 레이어

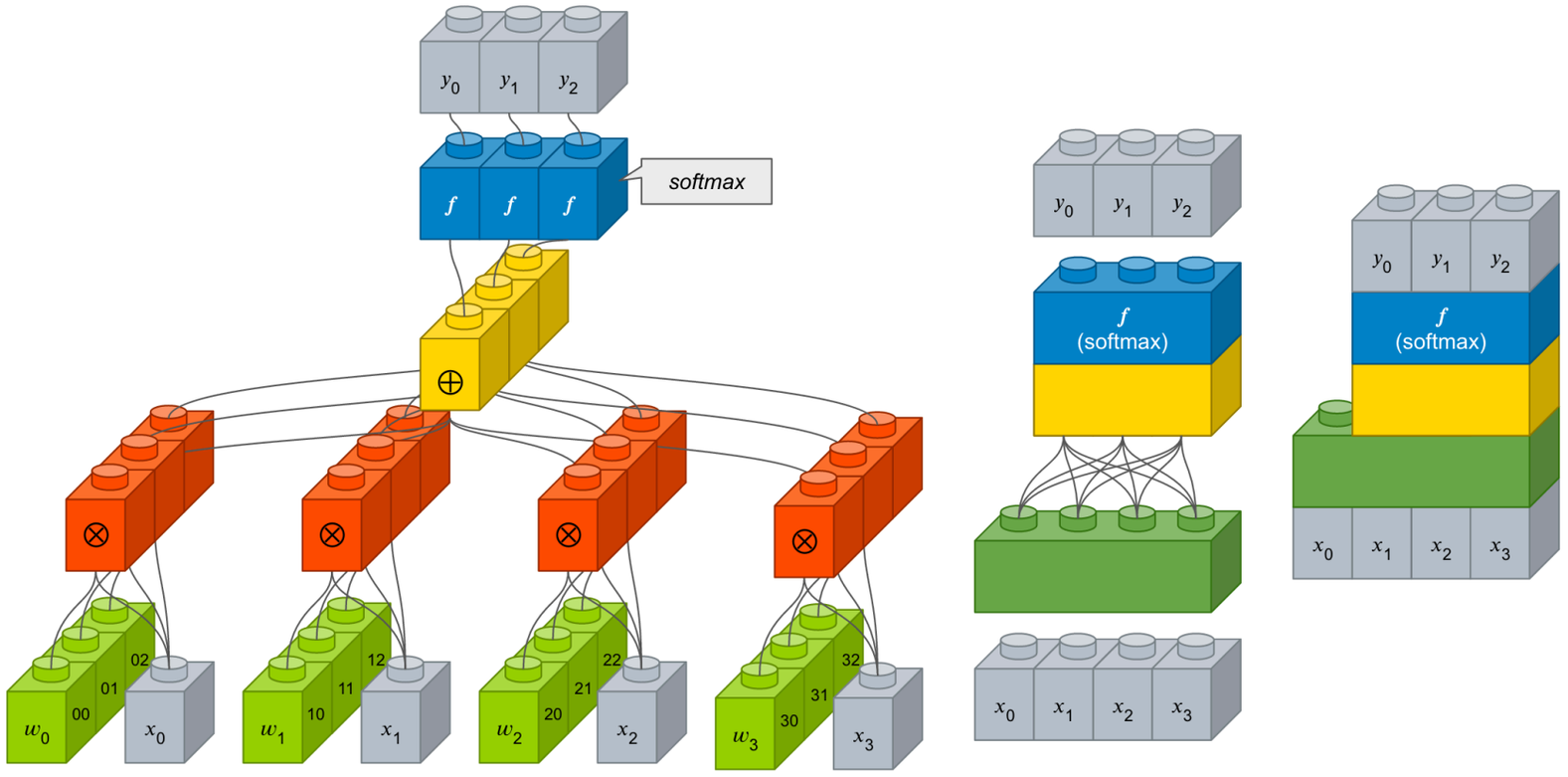
```
Dense(1, input_dim=3, activation='sigmoid'))
```



- 시그모이드 활성화 함수는 입력 뉴련과 가중치를 계산한 값이 0에서 1사이의 값이 나오기 때문에 이진 분류 문제에 적합합니다.

입출력을 모두 연결해주는 Dense 레이어

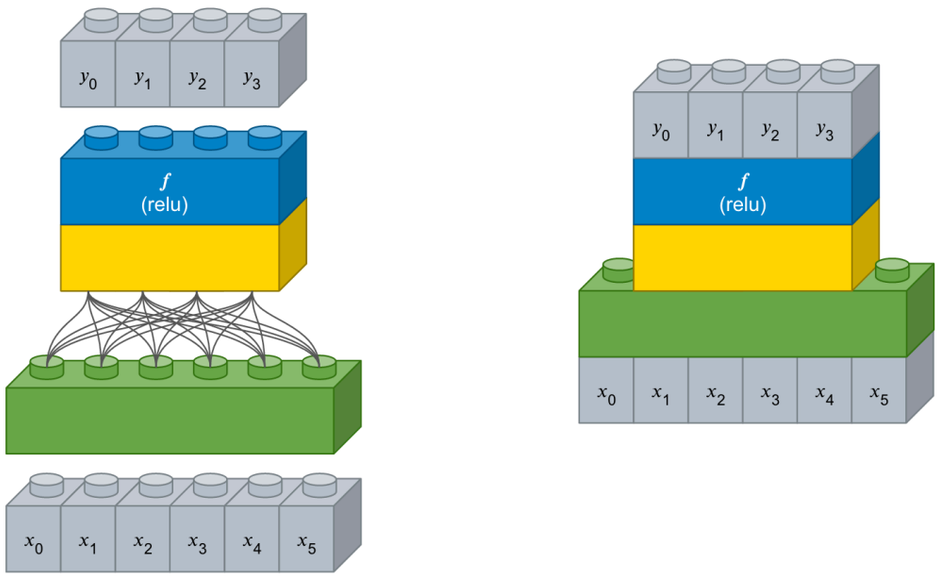
```
Dense(3, input_dim=4, activation='softmax'))
```



- 다중 클래스 분류 문제에서는 클래스 수 만큼의 출력이 필요합니다.
  입력 뉴련과 가중치를 계산한 값이 각 클래스의 확률 개념으로 표현할 수 있는 softmax 를 사용합니다. 입력 신호 4 * 출력 신호 3 = 12

입출력을 모두 연결해주는 Dense 레이어

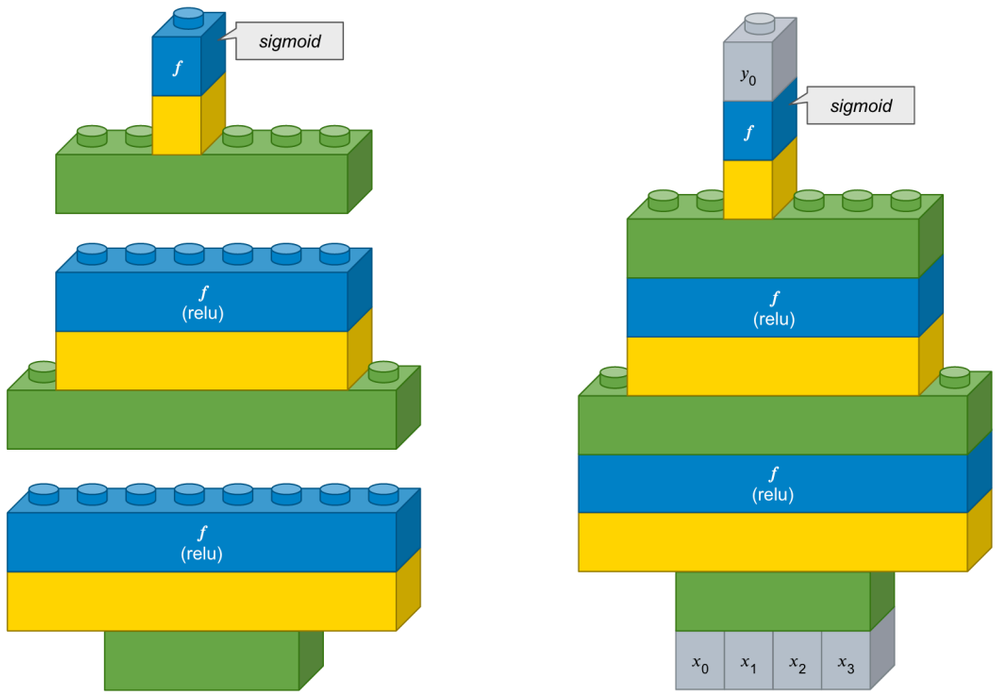
```
Dense(4, input_dim=6, activation='relu'))
```



- Dense Layer의 수치 입력, 은닉층일때 relu가 사용됩니다.
  relu는 역전파시에 성능이 좋은것으로 알려져 있습니다.

입출력을 모두 연결해주는 Dense 레이어

```
model.add(Dense(8, input_dim=4, init='uniform', activation='relu'))
model.add(Dense(6, init='uniform', activation='relu'))
model.add(Dense(1, init='uniform', activation='sigmoid'))
```



- 위의 코드에서 첫 Dense Layer 만이 입력 뉴런수를 설정해주고 이후의 Dense Layer 에서는 설정하지 않았습니다. 이전 계층으로 입력뉴런을 유추할 수 있기 때문입니다.

입출력을 모두 연결해주는 Dense 레이어

코드 시연

다층 퍼셉트론 모델 만들어보기

1.문제 정의하기
2.데이터 준비하기
3.데이터셋 생성하기
4.모델 구성하기
5.모델 학습과정 설정하기
6.모델 학습시키기
7.모델 평가하기

다층 퍼셉트론 모델 만들어보기

1. 문제 정의하기

8개의 변수와 발병 유무가 기록된 '파마족 인디언 당뇨병 발병 데이터셋'
이 데이터셋의 8개를 독립 변수로 보고 발병유무 예측을 이진 분류 문제로 정의

위의 데이터를 선정한 이유

- 인스턴스 수와 속성 수가 예제로 사용하기에 적당합니다.
- 모든 특징이 정수 혹은 실수로 되어 있어서 별도의 전처리 과정이 필요없습니다.

다층 퍼셉트론 모델 만들어보기

2. 데이터 준비하기

- 인스턴스 수 : 768개 => 컬럼 개수
- 속성 수 : 8가지 =>
- 클래스 수 : 2가지 => 발병, O X

 # 1. Number of times pregnant
 # 2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
 # 3. Diastolic blood pressure (mm Hg)
 # 4. Triceps skin fold thickness (mm)
 # 5. 2-Hour serum insulin (mu U/ml)
 # 6. Body mass index (weight in kg/(height in m)^2)
 # 7. Diabetes pedigree function
 # 8. Age (years)
 # 9. Class variable (0 or 1)

 # 1. 임신 횟수
 # 2. 경구 포도당 내성 검사에서 혈장 포도당 농도 2 시간
 # 3. 확장기 혈압 (mm Hg)
 # 4. 삼두근 피부 배 두께 (mm)
 # 5. 2 시간 혈청 인슐린 (mu U / ml)
 # 6. 체질량 지수 (체중 kg / (높이) ^ 2)
 # 7. 당뇨병 혈통 기능
 # 8. 나이 (세)
 # 9. 클래스 변수 (0 또는 1)

다층 퍼셉트론 모델 만들어보기

2. 데이터 준비하기

좀 더 살펴보면,
양성인 경우가 268개(34.9%),
음성인 경우가 500개(65.1%)입니다.
즉 모델이 모두 음성이라고 판별을 한다하더라도 65.1%의 기본 정확도(baseline accuracy)를 달성할 수 있습니다.
즉 우리의 모델이 65.1%보다 낮으면
모두 음성이라고 판별하는 것보다 낮은 정확도를 가진다고 생각하시면 됩니다.
지금까지 개발된 알고리즘의 최대 정확도는 10-fold 교차검증(cross validataion) 했을 때 77.7%이라고 웹사이트에는 표기되어 있습니다.

본 예제에서 우리는 적어도 65.1%보다 높은 정확도를 가진 모델을 만들어야 한다.

다층 퍼셉트론 모델 만들어보기

3. 데이터 생성하기

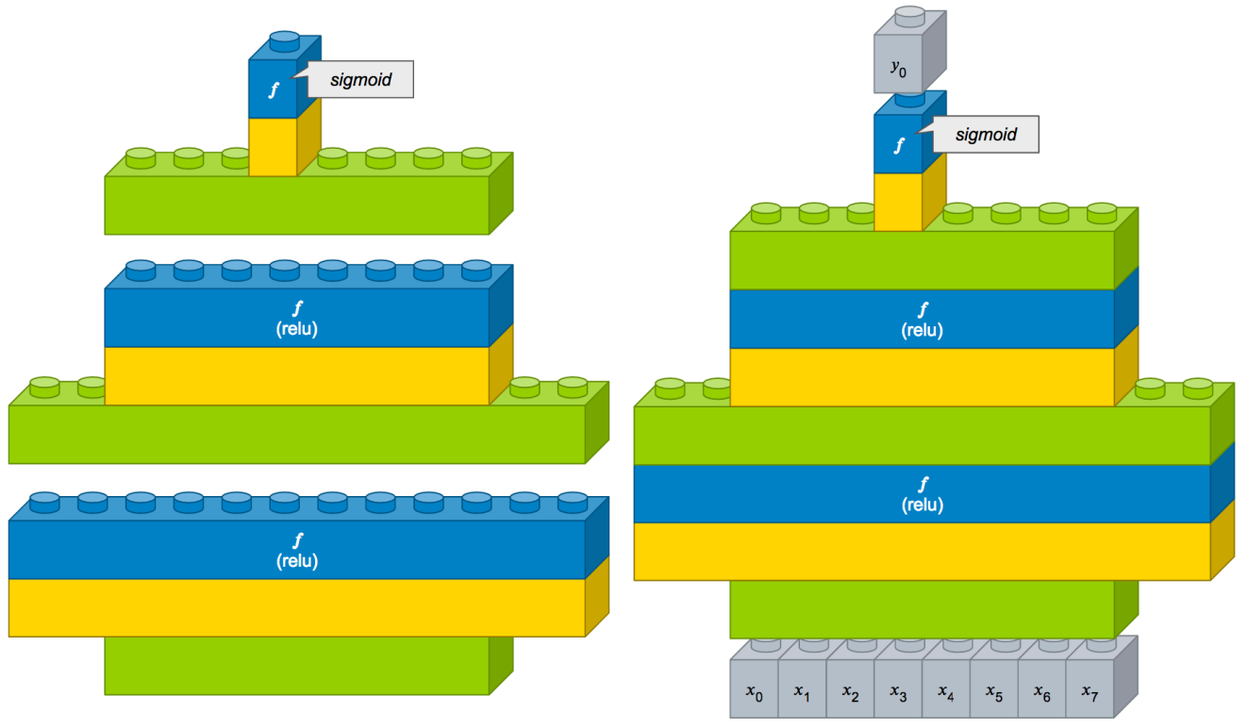
```
x_train = dataset[:700,0:8]
y_train = dataset[:700,8]
x_test = dataset[700:,0:8]
y_test = dataset[700:,8]
```

6,148,72,35,0,33.6,0.627,50,1

 # 1. Number of times pregnant
 # 2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
 # 3. Diastolic blood pressure (mm Hg)
 # 4. Triceps skin fold thickness (mm)
 # 5. 2-Hour serum insulin (mu U/ml)
 # 6. Body mass index (weight in kg/(height in m)^2)
 # 7. Diabetes pedigree function
 # 8. Age (years)
 # 9. Class variable (0 or 1)

다층 퍼셉트론 모델 만들어보기

4. 데이터 구성하기




```
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

다층 퍼셉트론 모델 만들어보기

5. 모델 학습과정 설정하기

```
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

- loss : 현재 가중치 세트를 평가하는 데 사용한 손실 함수 입니다. 이진 클래스 문제이므로 'binary_crossentropy'으로 지정합니다.

평균 제곱 계열	mean_squared_error	평균 제곱 오차
	mean_absolute_error	평균 절대 오차(실제 값과 예측 값 차이의 절
	mean_absolute_percentage_	평균 절대 백분율 오차(절댓값 오차를 절댓
	mean_squared_logarithmic_	평균 제곱 로그 오차(실제 값과 예측 값에 로
교차 엔트로피 계열	categorical_crossentropy	범주형 교차 엔트로피(일반적인 분류)
	binary_crossentropy	이항 교차 엔트로피(두 개의 클래스 중에서

- optimizer : 최적의 가중치를 검색하는 데 사용되는 최적화 알고리즘으로 효율적인 경사 하강법 알고리즘 중 하나인 'adam'을 사용합니다.
  http://dalpo0814.tistory.com/29
- metrics : 평가 척도를 나타내며 분류 문제에서는 일반적으로 'accuracy'으로 지정합니다. 다중 클래스 분류 문제에서는 matrics를 확장하여 사용
  https://tykimos.github.io/2017/09/24/Custom_Metric/

다층 퍼셉트론 모델 만들어보기

6. 모델 학습시키기

```
model.fit(x_train, y_train, epochs=1500, batch_size=64)
```

- 첫번째 인자 : 입력 변수입니다. 8개의 속성 값을 담고 있는 X를 입력합니다.
- 두번째 인자 : 출력 변수 즉 라벨값입니다. 결과 값을 담고 있는 Y를 입력합니다.
- epochs : 전체 훈련 데이터셋에 대해 학습 반복 횟수를 지정합니다.
- batch_size : 가중치를 업데이트할 배치 크기를 의미하며, 64개로 지정했습니다.

다층 퍼셉트론 모델 만들어보기

7. 모델 평가하기

```
scores = model.evaluate(x_test, y_test)
print("%s: %.2f%%" %(model.metrics_names[1], scores[1]*100))
```

```
67/67 [==============================] - 0s 1ms/step
acc: 82.09%
```

다층 퍼셉트론 모델 만들어보기

코드 시연