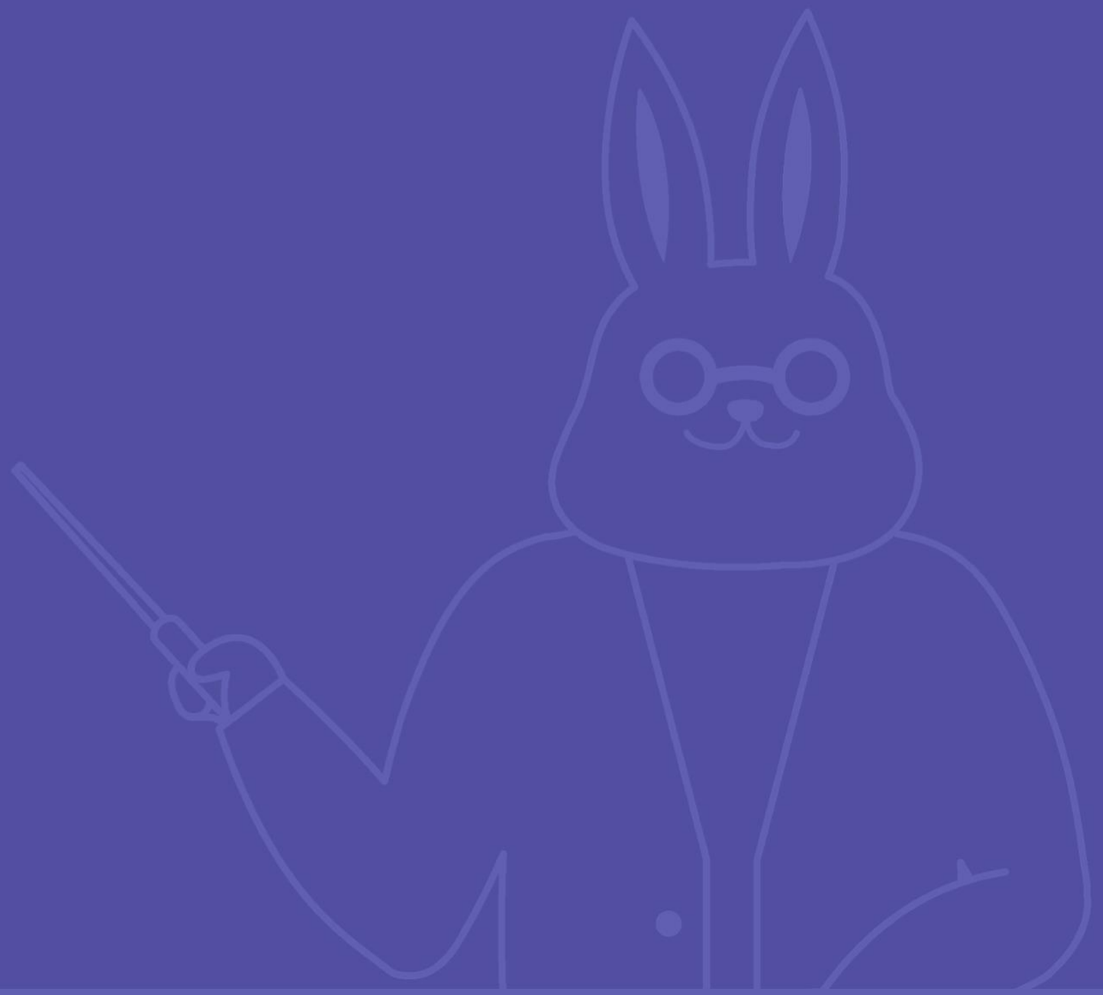
딥러닝 시작하기

02 텐서플로우와 신경망



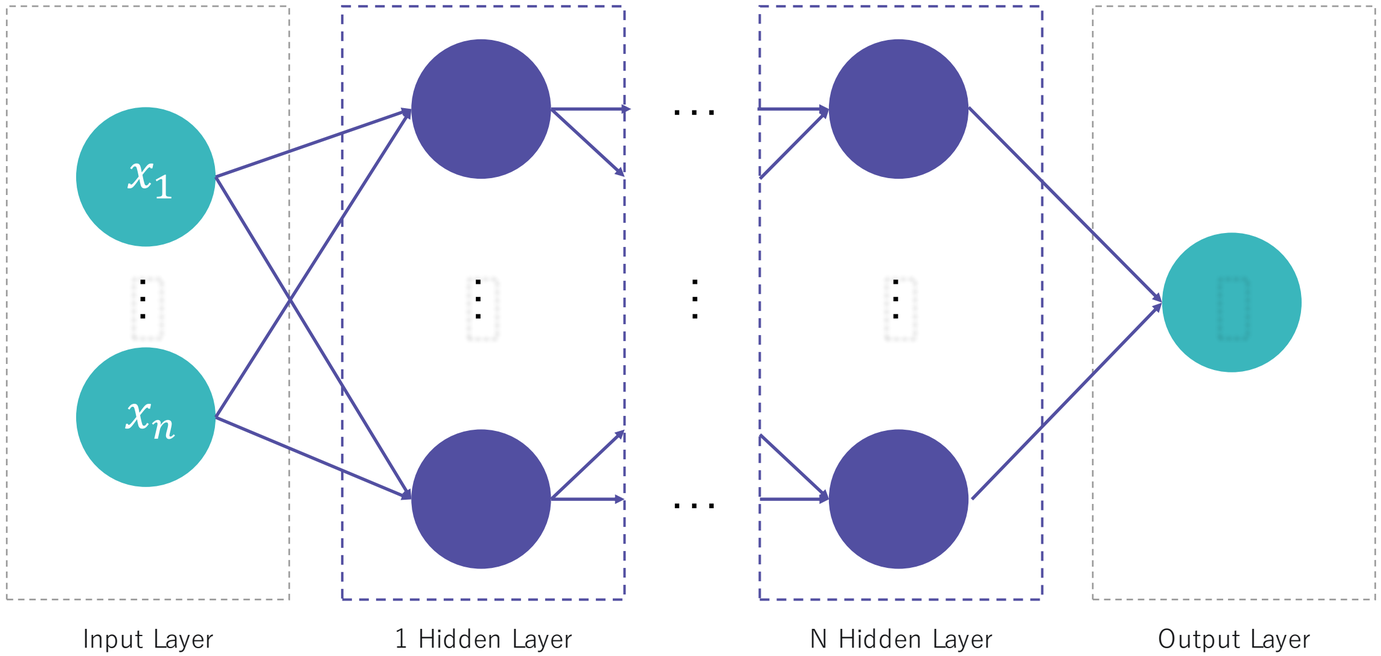
/* elice */

목차

01

딥러닝 모델의 학습 방법

딥러닝 모델이란



히든층이 많아진다면,

깊은 신경망이라는 의미의 Deep Learning 단어 사용

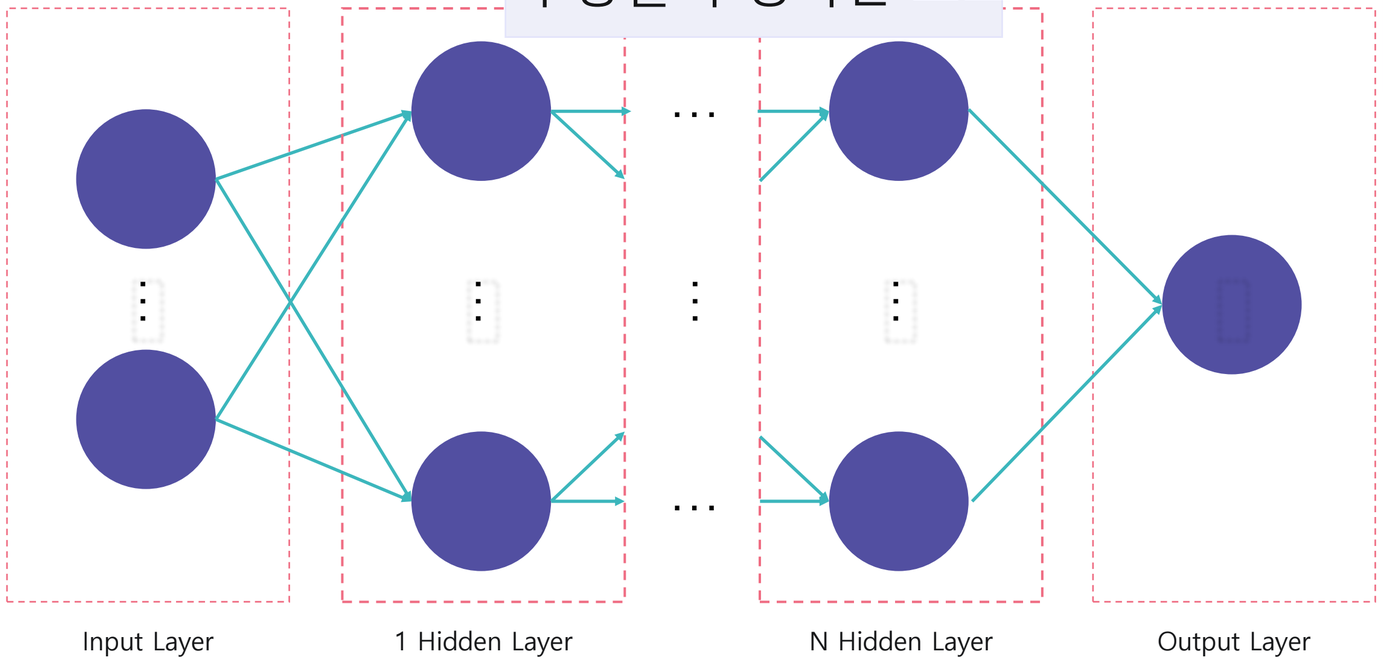
딥러닝 모델의 구성 요소

Node/Unit

각 층을 구성하는 요소

Layer

모델을 구성하는 층



가중치(Weight)

노드간의 연결강도

딥러닝 모델의 학습 방법

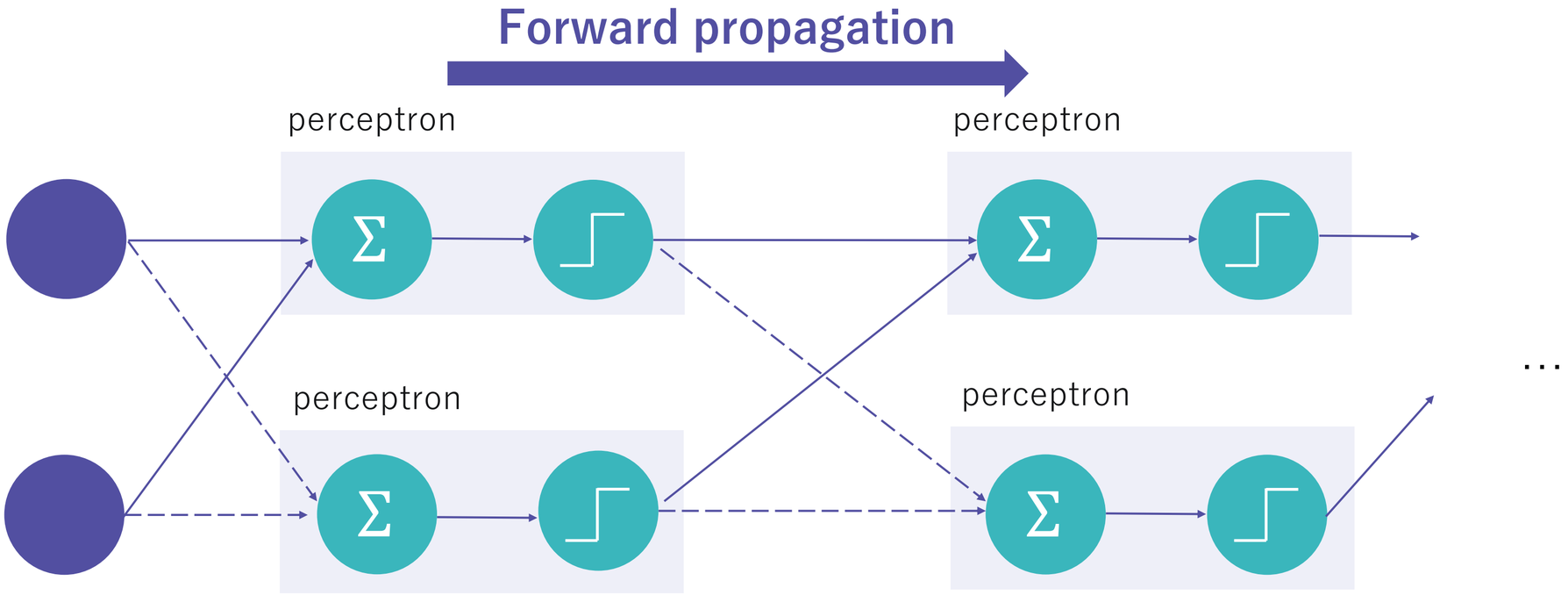
예측값과 **실제값** 간의 **오차값**을 **최소화**하기 위해

오차값을 **최소화**하는 모델의 인자를 찾는 알고리즘을 적용

Loss function을 최소화하는 **가중치**를 찾기 위해 **최적화** 알고리즘을 적용

우선 딥러닝 모델에서 예측값을 구하는 방법을 알아보자

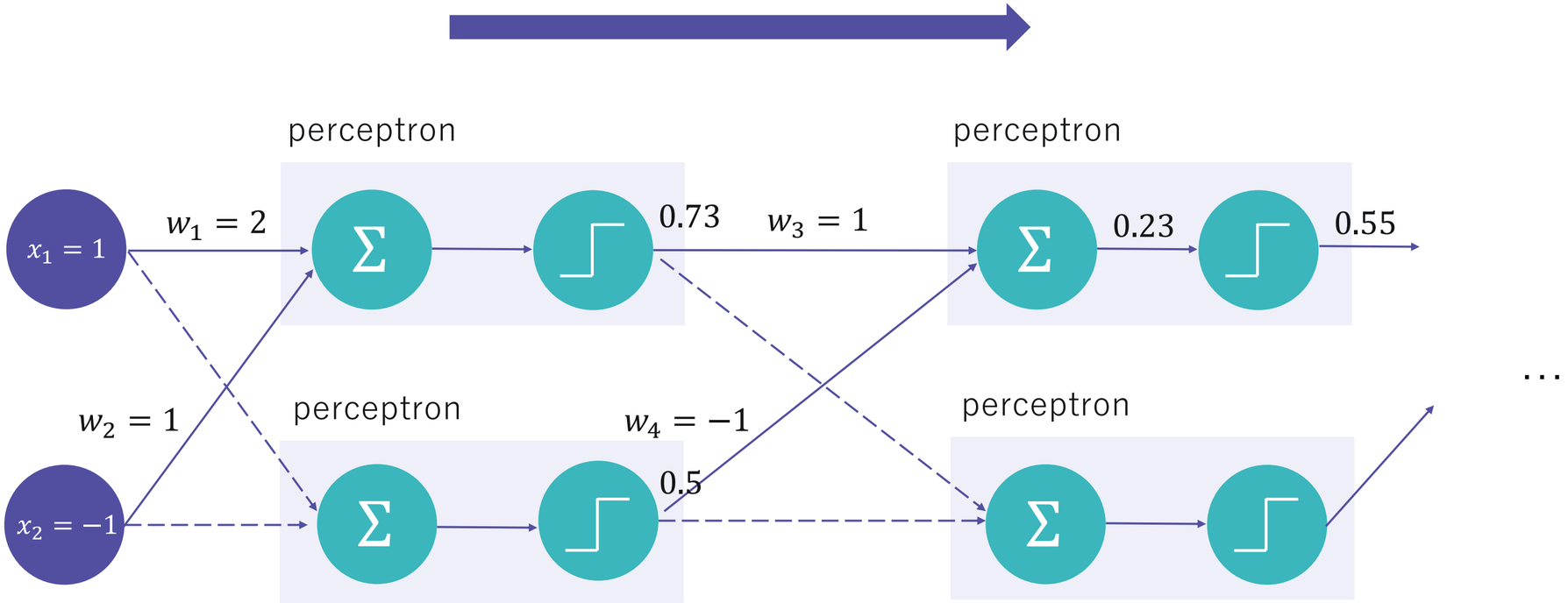
딥러닝 모델이 예측값 구하는 방식



순전파(Forward propagation)

입력 값을 바탕으로 출력 값을 계산하는 과정

순전파 예시


perceptron
perceptron
$x_1 = 1$
$w_1 = 2$
0.73
$w_3 = 1$
0.23
0.55
$w_2 = 1$
perceptron
$w_4 = -1$
perceptron
0.5
$x_2 = -1$
...

최적화 방식 살펴보기

순전파를 사용하면 **예측값과 실제값 간의 오차값**을 구하여

Loss function을 구할 수 있음

그렇다면 어떻게 최적화를 해야할까?

=> **경사 하강법(Gradient descent)을 사용**

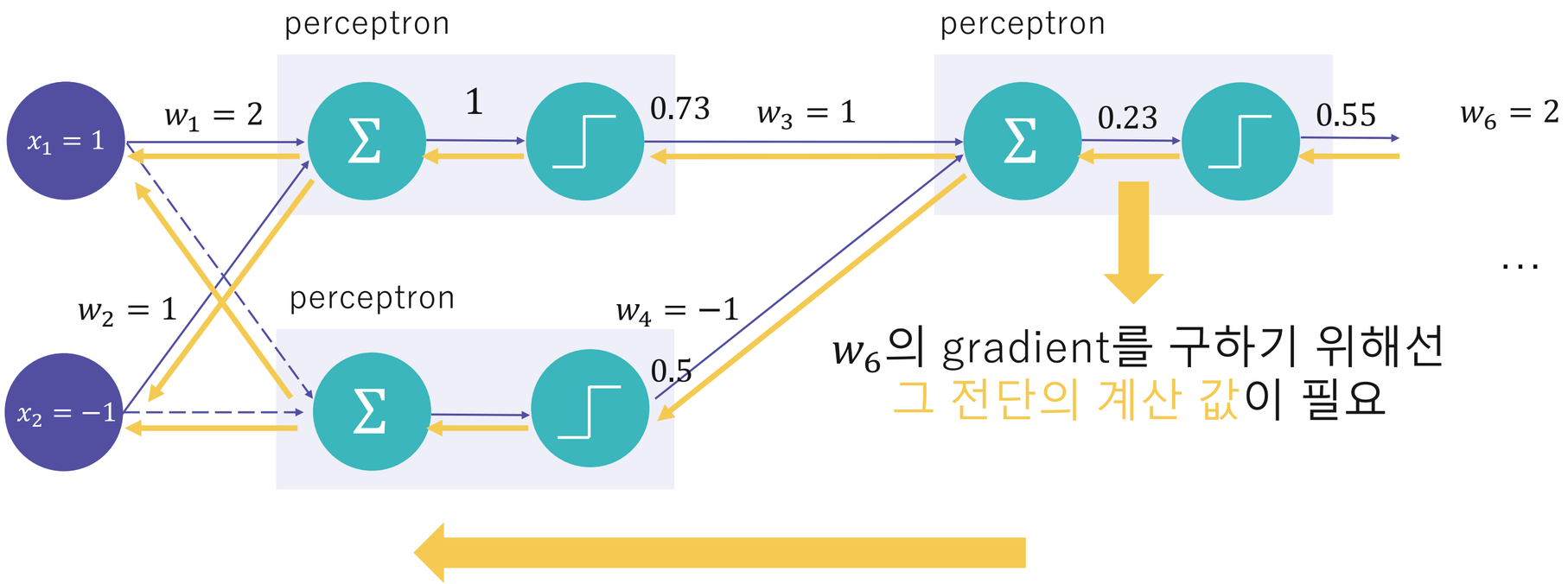
경사 하강법(Gradient descent)

가중치를 **Loss function** 값이 작아지게 업데이트 하는 방법

가중치는 **Gradient 값**을 사용하여 업데이트를 수행함

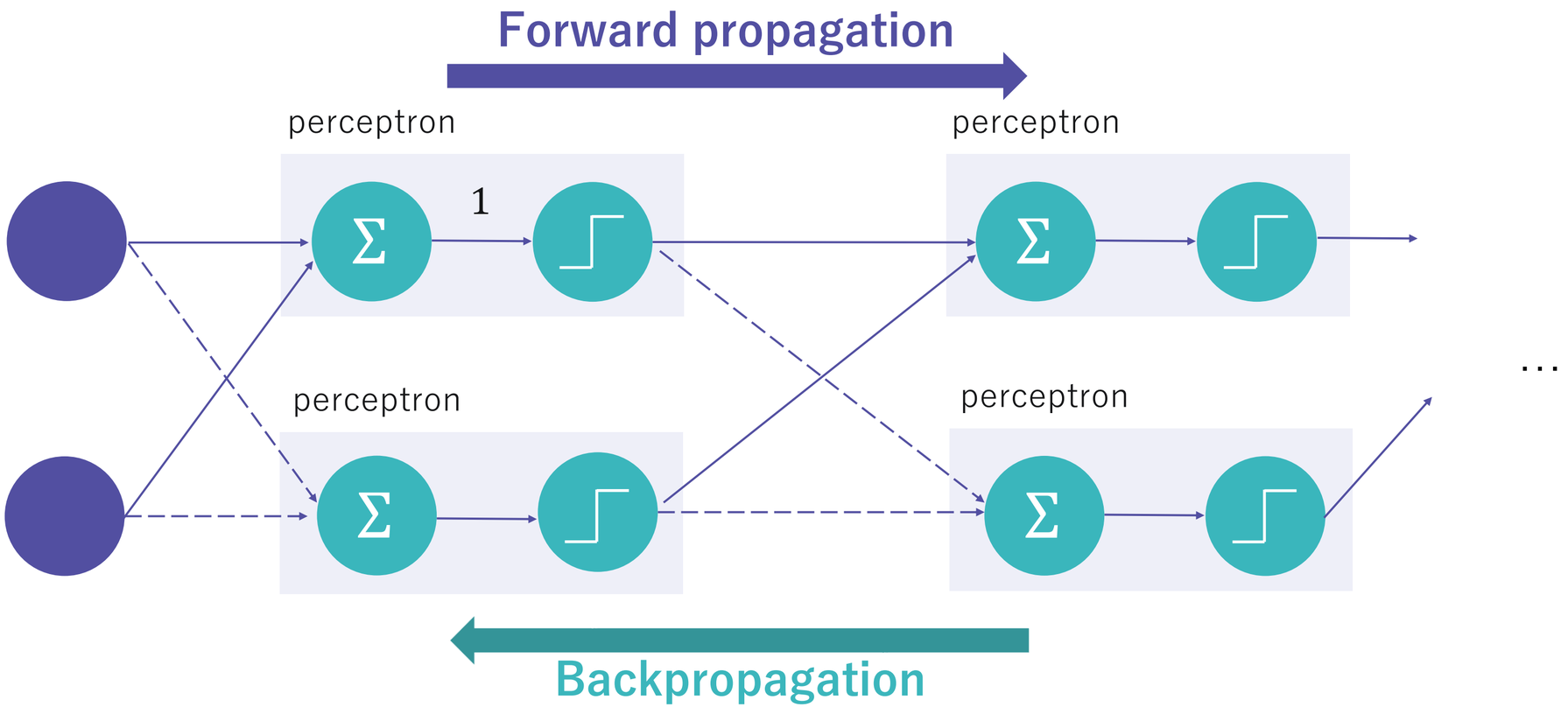
Gradient 값은 각 가중치 마다 정해지며,
역전파(Backpropogation)를 통하여 구할 수 있음

역전파(Backpropogation)



Forward propagation의 반대 방향으로 이루어지는 과정

가중치 업데이트 과정



위 과정을 수행하여 **가중치**들을 업데이트 할 수 있으며, 이를 반복하여

Loss function을 제일 작게 만드는 **가중치**를 구함

딥러닝 모델의 학습 방법 정리

딥러닝 모델의 학습 순서

1. 학습용 feature 데이터를 입력하여 예측값 구하기 **(순전파)**
2. 예측값과 실제값 사이의 오차 구하기 **(Loss 계산)**
3. Loss를 줄일 수 있는 가중치 업데이트 하기 **(역전파)**
4. 1~3번 반복으로 Loss를 최소로 하는 가중치 얻기

02

텐서플로우로 딥러닝 구현하기
- 데이터 전 처리

텐서플로우(TensorFlow)

TensorFlow

유연하고, 효율적이며, 확장성 있는 딥러닝 프레임워크

대형 클러스터 컴퓨터부터 스마트폰까지 다양한 디바이스에서 동작 가능

딥러닝 모델 구현 순서

1. 데이터 전 처리하기
2. 딥러닝 모델 구축하기
3. 모델 학습시키기
4. 평가 및 예측하기

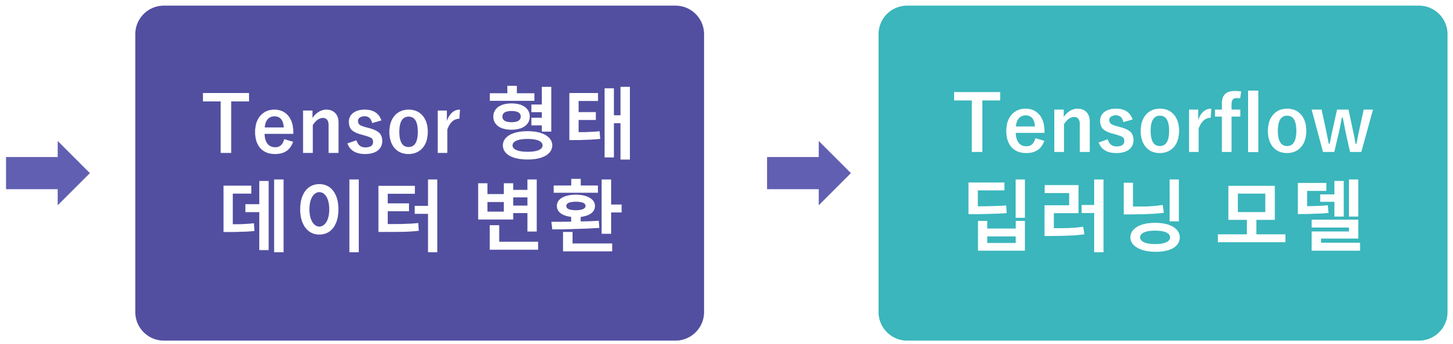
1. 데이터 전 처리하기

Tensorflow 딥러닝 모델은 **Tensor 형태의 데이터**를 입력 받는다.

Tensor란 다차원 배열로서 tensorflow에서 사용하는 객체

데이터

	age	sex	cp	trestbps	chol	fbs	restecg	thalach
0	63	1	1	145	233	1	2	150
1	67	1	4	160	286	0	2	108
2	67	1	4	120	229	0	2	129
3	37	1	3	130	250	0	0	187
4	41	0	2	130	204	0	2	172



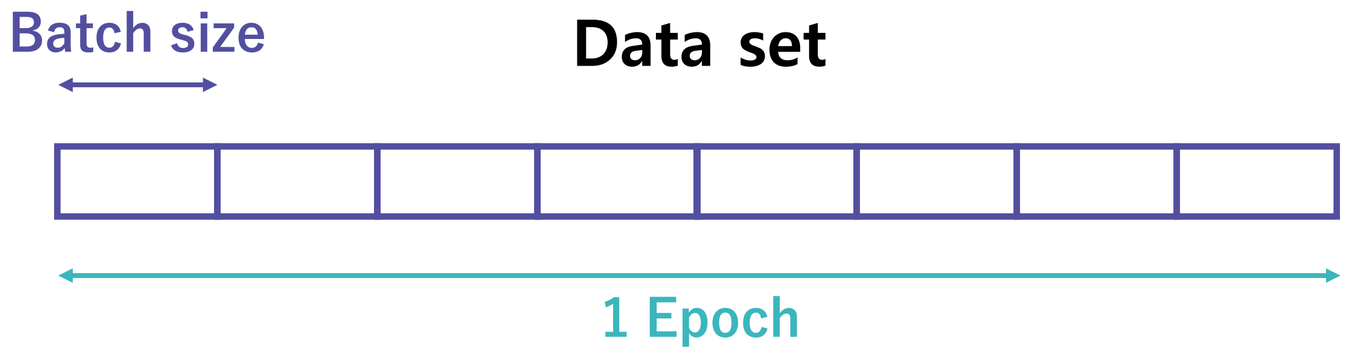
1. 데이터 전 처리하기: tf.data.Dataset

```
# pandas를 사용하여 데이터 불러오기
df = pd.read_csv('data.csv')
feature = df.drop(columns=['label'])
label = df['label']


# tensor 형태로 데이터 변환
dataset = tf.data.Dataset.from_tensor_slices((feature.values, label.values))
```

Dataset API를 사용하여 딥러닝 모델 용 Dataset을 생성

1. 데이터 전 처리하기: Epoch와 Batch



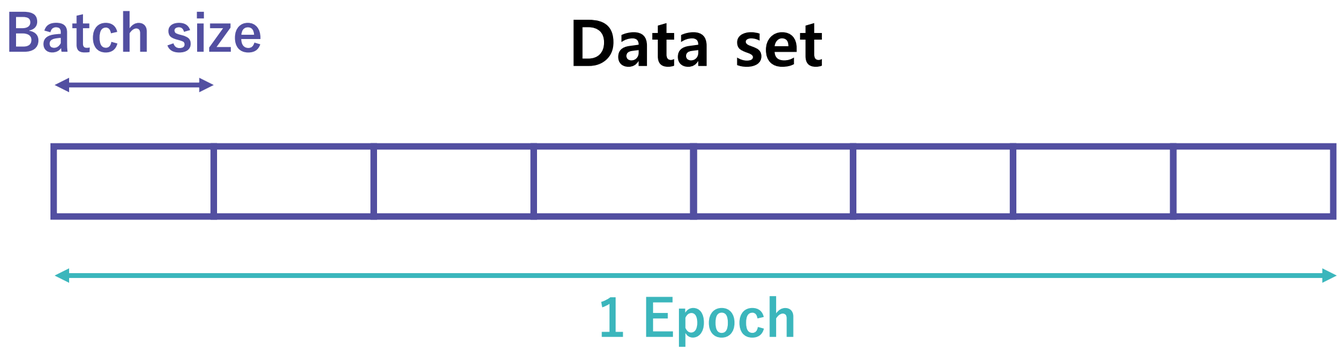
딥러닝에 사용하는 데이터는 추가 적인 전 처리 작업이 필요함 => **Epoch, Batch**

Epoch: 한 번의 epoch는 전체 데이터 셋에 대해 **한 번 학습을 완료한 상태**

Batch: 나눠진 데이터 셋 (보통 mini-batch라고 표현)

iteration는 epoch를 나누어서 실행하는 횟수를 의미

1. 데이터 전 처리하기: Epoch와 Batch 예시



Ex) 총 데이터가 1000개, Batch size = 100

- 1 iteration = 100개 데이터에 대해서 학습
- 1 epoch = 100/Batch size = 10 iteration

1. 데이터 전 처리하기: tf.data.Dataset

```
# tensor 형태로 데이터 변환
dataset = tf.data.Dataset.from_tensor_slices((feature.values, label.values))

# dataset의 batch 사이즈를 32로 설정
dataset = dataset.batch(32)
```

03

텐서플로우로 딥러닝 구현하기 - 모델 구현

2. 딥러닝 모델 구축하기 : 고수준 API 활용

텐서플로우의 패키지로 제공되는 고수준 API

딥러닝 모델을 간단하고 빠르게 구현 가능

딥러닝 모델 구축을 위한 Keras 메소드(1)

모델 클래스 객체 생성

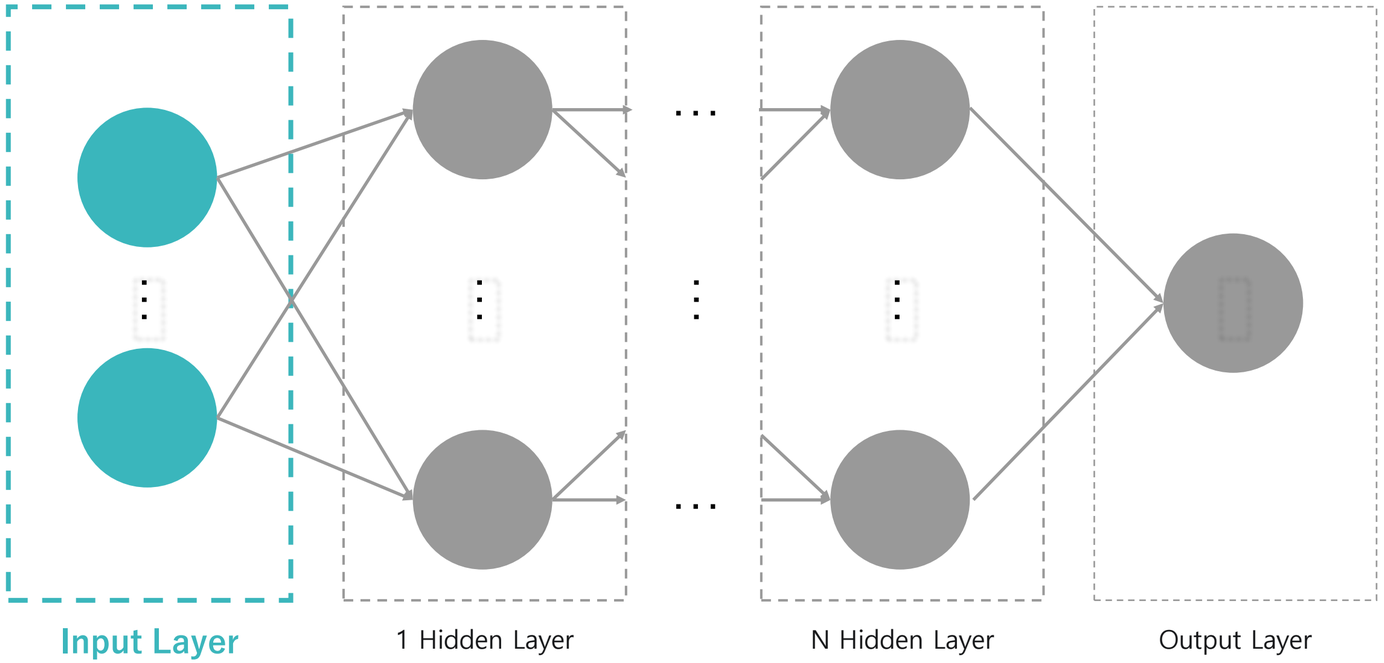
```
tf.keras.models.Sequential()
```

모델의 각 Layer 구성

```
tf.keras.layers.Dense(units, activation)
```

- units : 레이어 안의 Node의 수
- activation : 적용할 activation 함수 설정

Input Layer의 입력 형태 지정하기



첫 번째 즉, Input Layer는 입력 형태에 대한 정보를 필요로 함

input_shape / **input_dim** 인자 설정하기

모델 구축하기 코드 예시(1)

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(10, input_dim=2, activation='sigmoid'), # 2개의 입력 변수, 10개의 노드
    tf.keras.layers.Dense(10, activation='sigmoid'), # 10개의 노드
    tf.keras.layers.Dense(1, activation='sigmoid'), # 1개의 노드
])
```

딥러닝 모델 구축을 위한 Keras 메소드(2)

모델에 Layer 추가하기

```
[model].add(tf.keras.layers.Dense(units, activation))
```

- units : 레이어 안의 Node의 수
- activation : 적용할 activation 함수 설정

모델 구축하기 코드 예시(2)

```
model = tf.keras.models.Sequential()


model.add(tf.keras.layers.Dense(10, input_dim=2, activation='sigmoid'))
model.add(tf.keras.layers.Dense(10, activation='sigmoid'))
model.add(tf.keras.layers.Dense(1, activation='sigmoid'))
```

3. 딥러닝 모델 학습시키기 : Keras 메소드

모델 학습 방식을 설정하기 위한 함수

```
[model].compile(optimizer, loss)
```

- optimizer : 모델 학습 최적화 방법
- loss : 손실 함수 설정

모델을 학습시키기 위한 함수

```
[model].fit(x, y)
```

- x : 학습 데이터
- y : 학습 데이터의 label

딥러닝 모델 학습시키기 코드 예시

```
# MSE를 loss로 설정, 최적화 방식은 SGD 사용
model.compile(loss='mean_squared_error’, optimizer=‘SGD’)
# dataset에 저장된 데이터를 입력하고, epochs를 100으로 설정하고 학습
model.fit(dataset, epochs=100)
```

4. 평가 및 예측하기 : Keras 메소드

모델을 평가하기 위한 메소드

```
[model].evaluate(x, y)
```

- x: 테스트 데이터
- y:테스트 데이터의 label

모델로 예측을 수행하기 위한 함수

```
[model].predict(x)
```

- x : 예측하고자 하는 데이터

딥러닝 모델 학습시키기 코드 예시

```
# MSE를 loss로 설정, 최적화 방식은 SGD 사용
model.compile(loss='mean_squared_error’, optimizer=‘SGD’)
# dataset에 저장된 데이터를 입력하고, epochs를 100으로 설정하고 학습
model.fit(dataset, epochs=100)


# 모델 평가 및 예측하기
model.evaluate(X_test, Y_test)
predicted_labels_test = model.predict(X_test)
```