기계학습 활용 (10주차)

2019. 11. 15.

Prof. Seung Ho Lee

KOREATECH
한국기술교육대학교

강의 주제 : 딥러닝 활용(모델 설계)

- 이론
  - 딥러닝 모델 설계하기

- 실습
  - 딥러닝 모델 설계하기 (폐암 수술 환자의 생존율 예측하기)

지난 강의 요약

- 퍼셉트론Perceptron
  - 딥러닝의 기본 구성요소인 퍼셉트론의 동작 원리 이해
  - 퍼셉트론을 다층 구조로 쌓음으로써 XOR 문제와 같이 선형적으로 해결할 수 없는 문제 해결을 보임
  - 다층 퍼셉트론의 학습 원리 소개
    - ✓ 경사 하강법에 기반한 오차 역전파 알고리즘

출처 : 모두의 딥러닝(조태호)

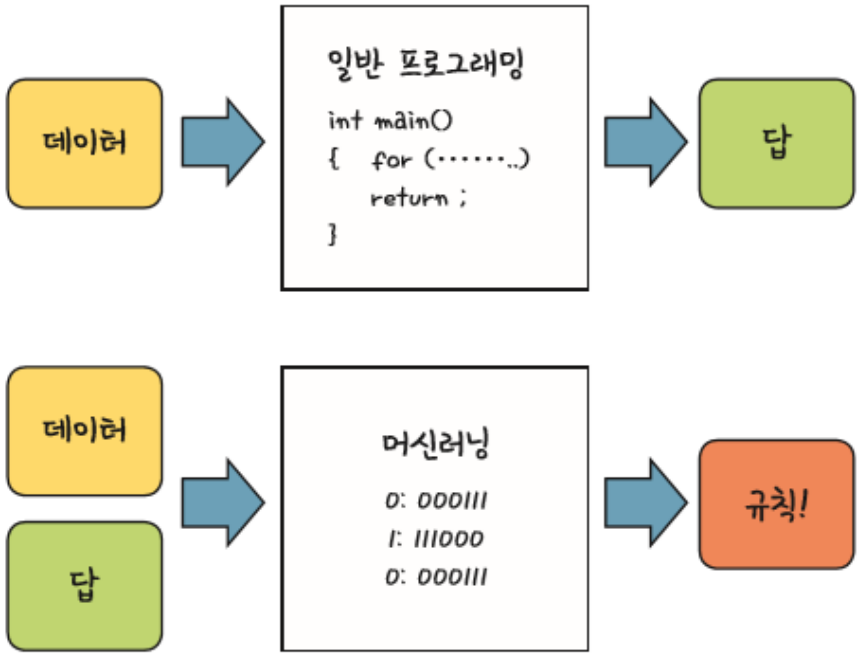
오늘 강의 요약

- 목표
  - 딥러닝의 신경망 구조 이해
  - 실제 활용할 데이터 셋 형태 이해
  - 파이썬과 케라스로 딥러닝의 신경망 모델 생성 및 평가

출처 : 모두의 딥러닝(조태호)

머신러닝과 예측

- **문제**
  - **"기존 환자의 데이터를 이용하여 새로운 환자의 생사를 예측하는 프로그램을 짜보아라."**

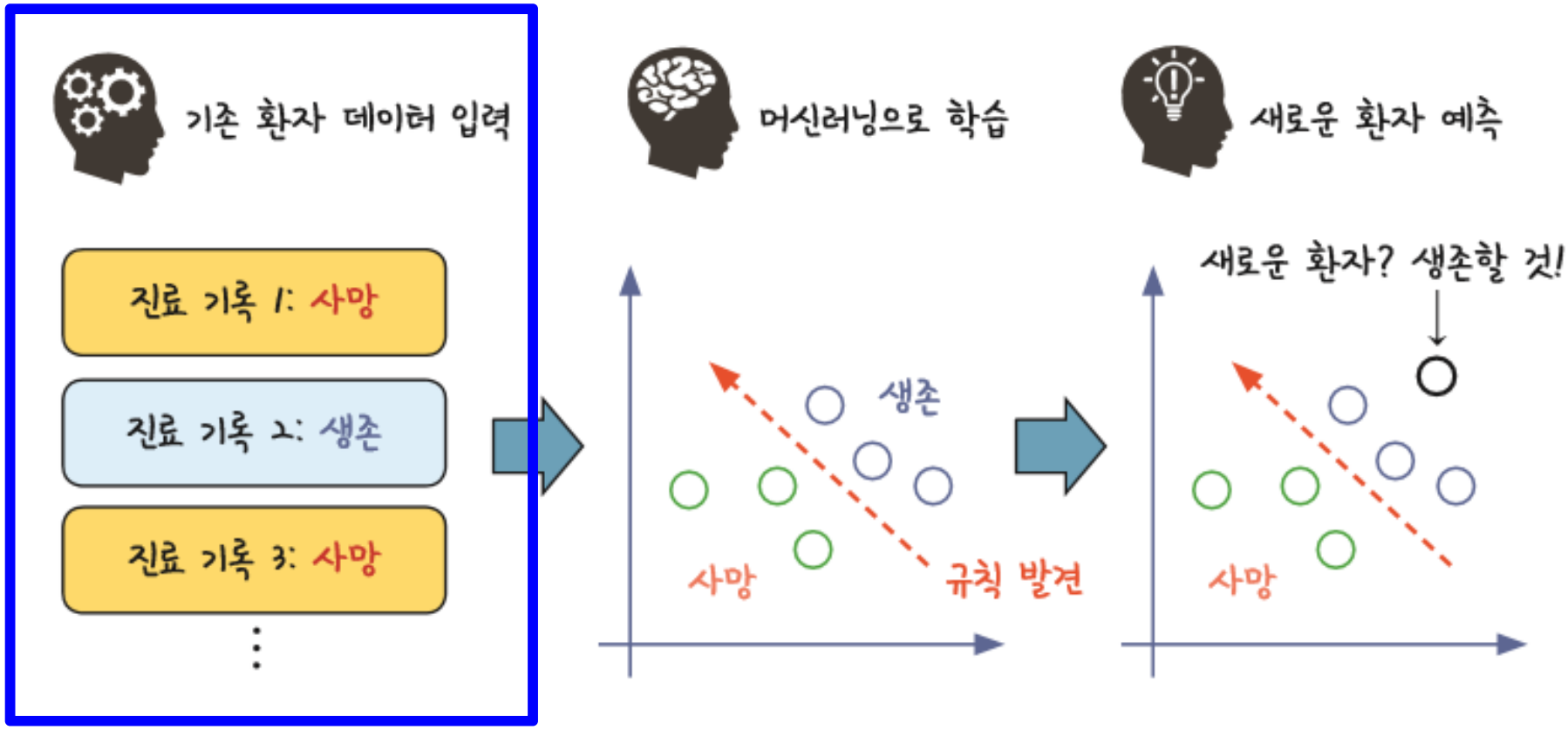


- **일반 프로그래밍 :** 규칙을 사람이 직접 입력하여 답을 구한다.
- **머신러닝 :** 데이터와 답을 입력해서 **규칙**을 찾음. 이 규칙을 **새로운 데이터에 적용!** (예측)

출처 : 모두의 딥러닝(조태호)

머신러닝과 예측

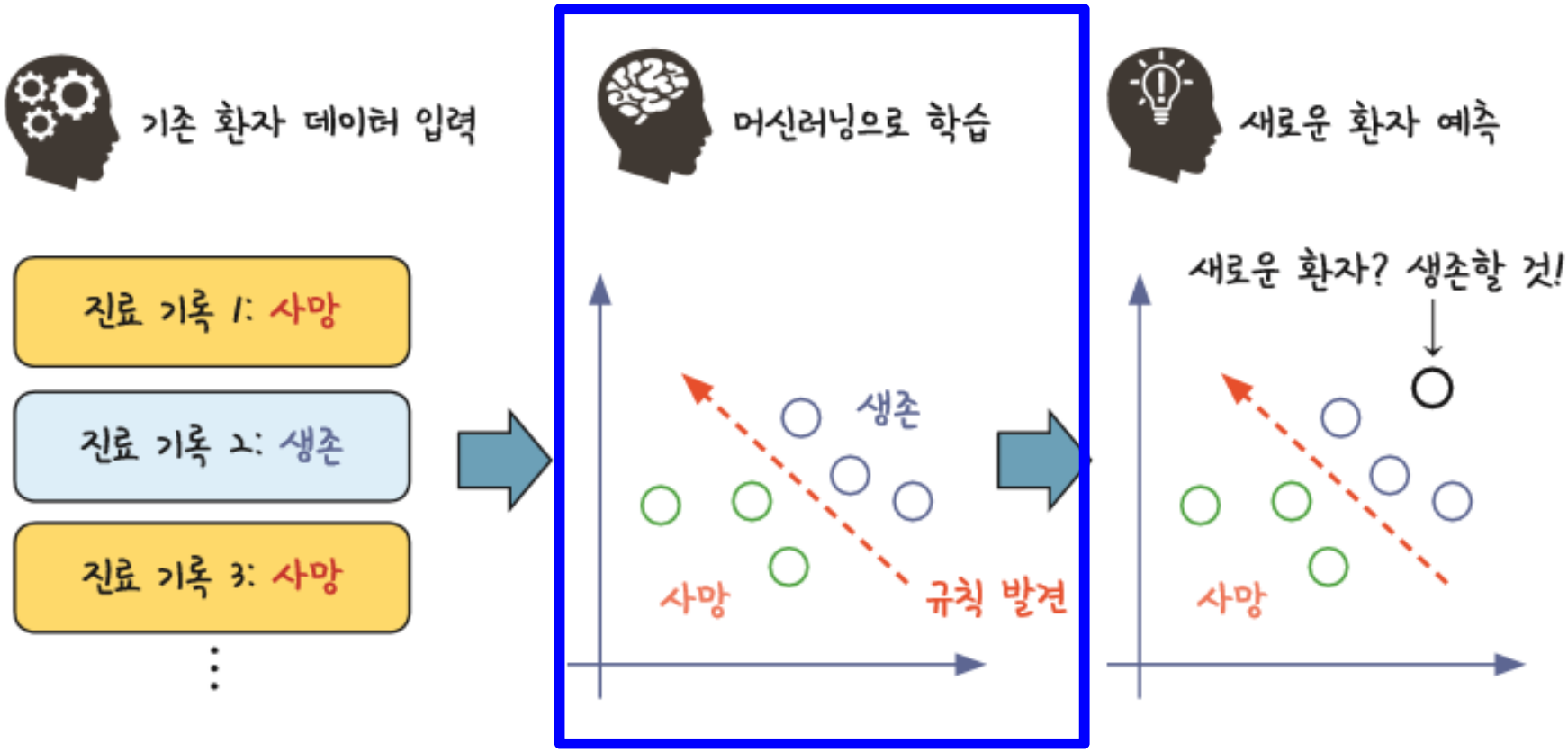
- 머신러닝 적용 순서
  - 의사가 그 동안 집도한 수술 환자의 수술 전 상태(=속성)와 수술 후의 생존여부(=정답)를 포함한 데이터를 준비



출처 : 모두의 딥러닝(조태호)

머신러닝과 예측

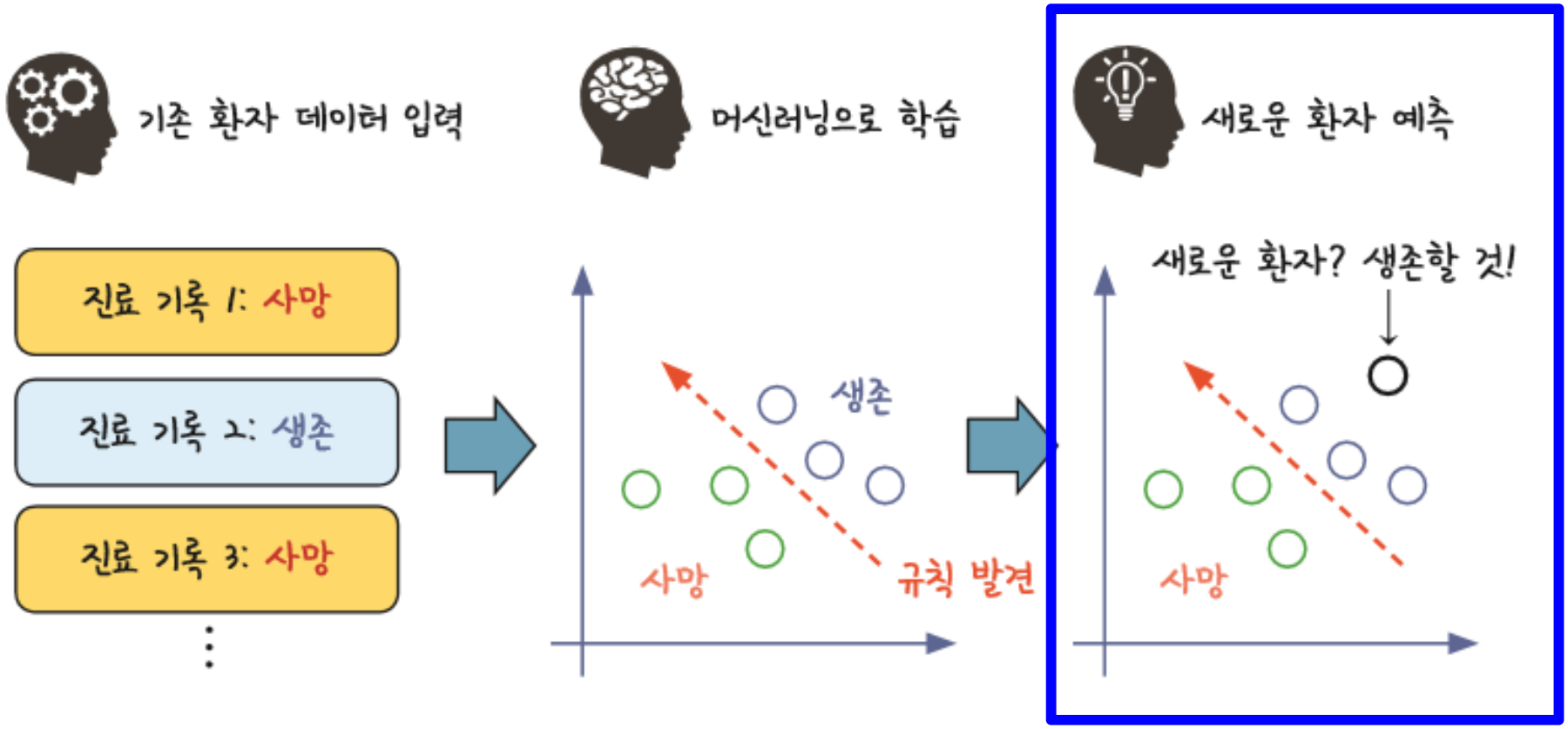
- 머신러닝 적용 순서
  - 기존 환자의 데이터가 머신러닝 알고리즘에 입력되는 순간 규칙 (=패턴)이 분석됨



출처 : 모두의 딥러닝(조태호)

머신러닝과 예측

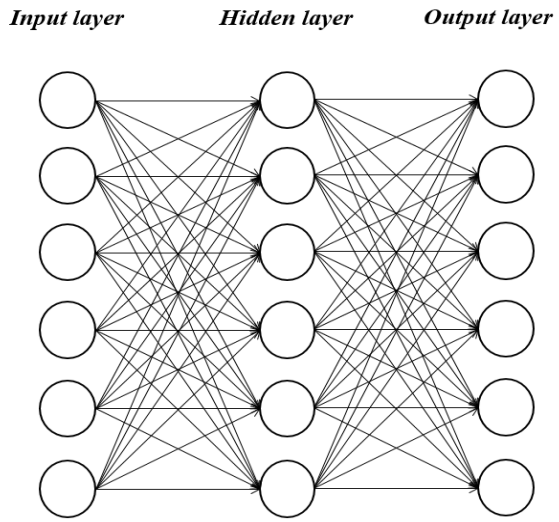
- 머신러닝 적용 순서
  - 분석된 규칙을 새로운 환자 데이터에 적용하여 생존 여부 예측



출처 : 모두의 딥러닝(조태호)

딥러닝 모델

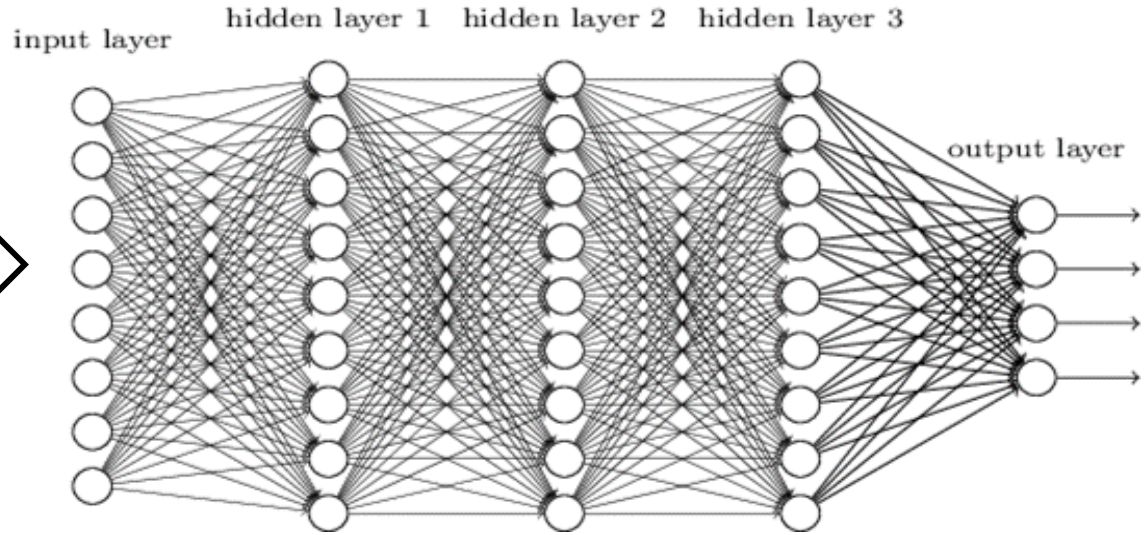
0층 입력층 | 1층 은닉층 | 2층 출력층



Neural Network (신경망)

더 깊게

0층 입력층 | 1층 은닉층 | 2층 은닉층 | 3층 은닉층 | 4층 출력층



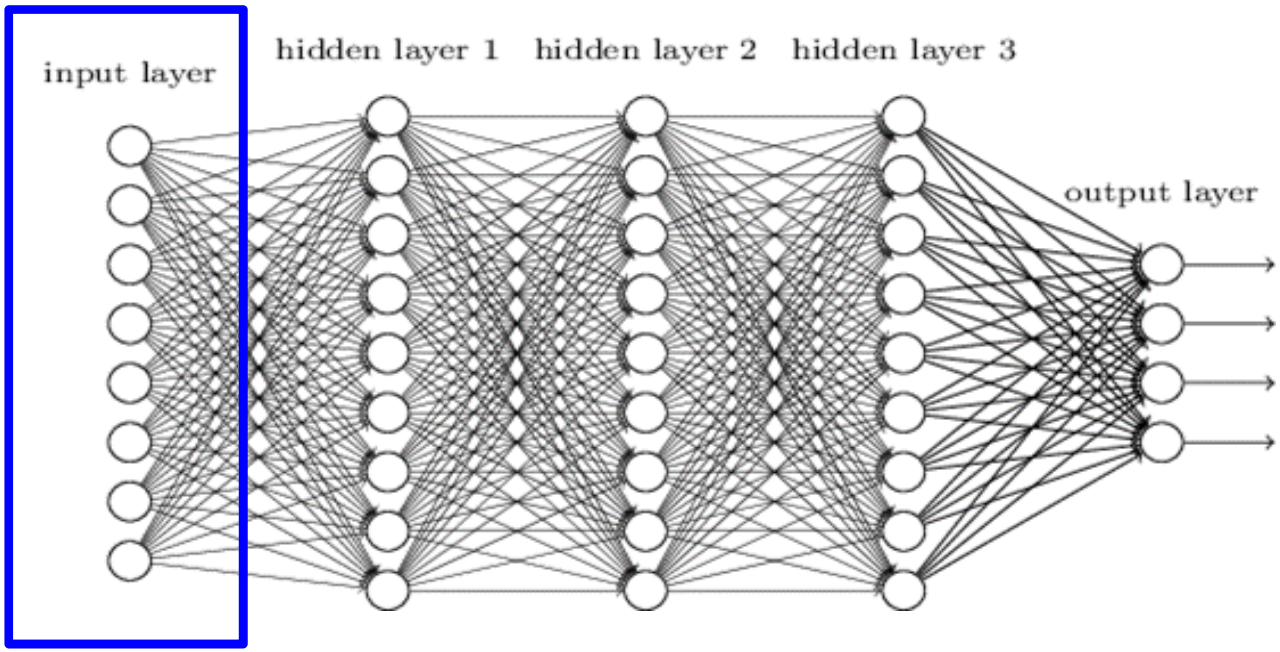
Deep Neural Network (**심층** 신경망)

➔ 심층 신경망을 학습하는 것이 딥러닝

◯ : 노드 또는 뉴런이라고 함

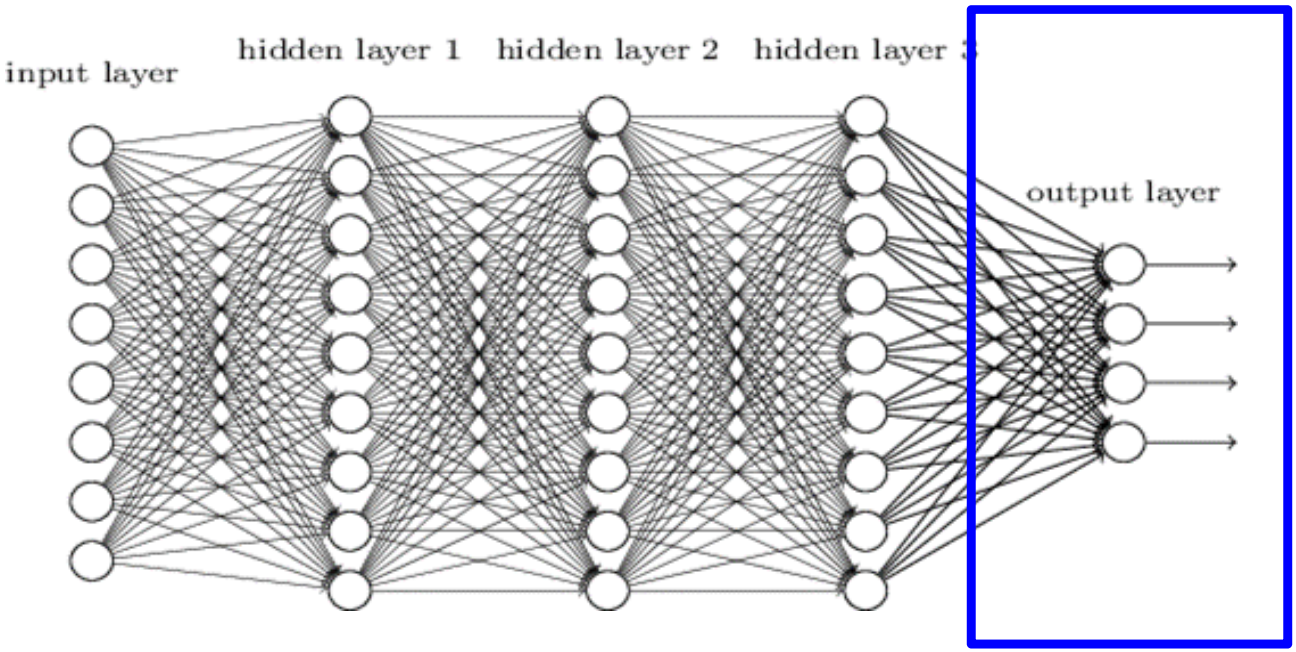
딥러닝 모델 구조 설계

- 입력층 노드 개수 설정
  - 데이터 속성 개수에 의해 결정됨



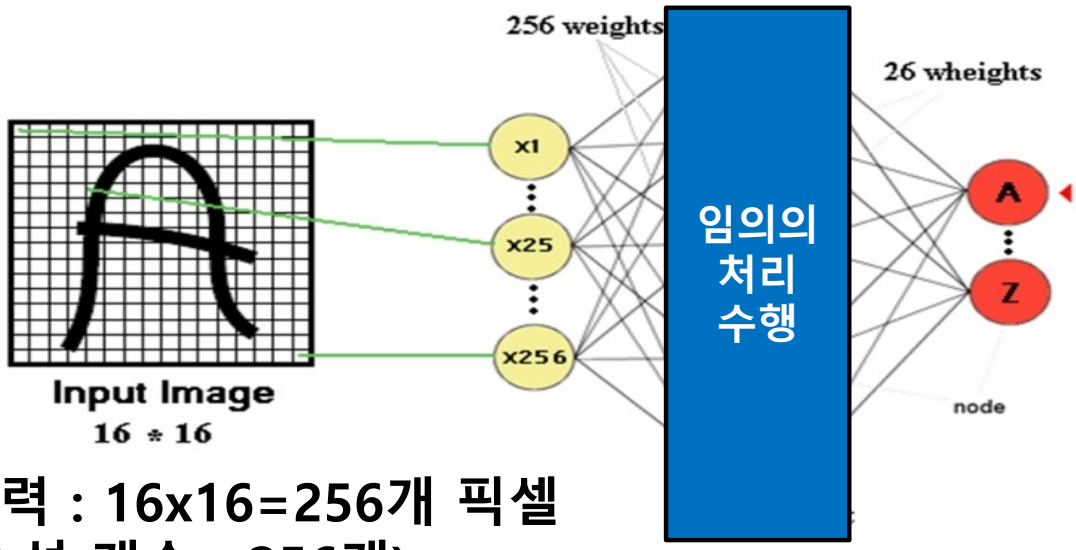
딥러닝 모델 구조 설계

- **출력층 노드 개수 설정**
  - 회귀 : 1개
  - 이진 분류(클래스 2개) : 1개
  - 다중 분류(클래스 3개 이상) : 클래스가 N개이면 노드도 N개



딥러닝 모델 구조 설계

- 입력층, 출력층 노드 개수 설정 예시



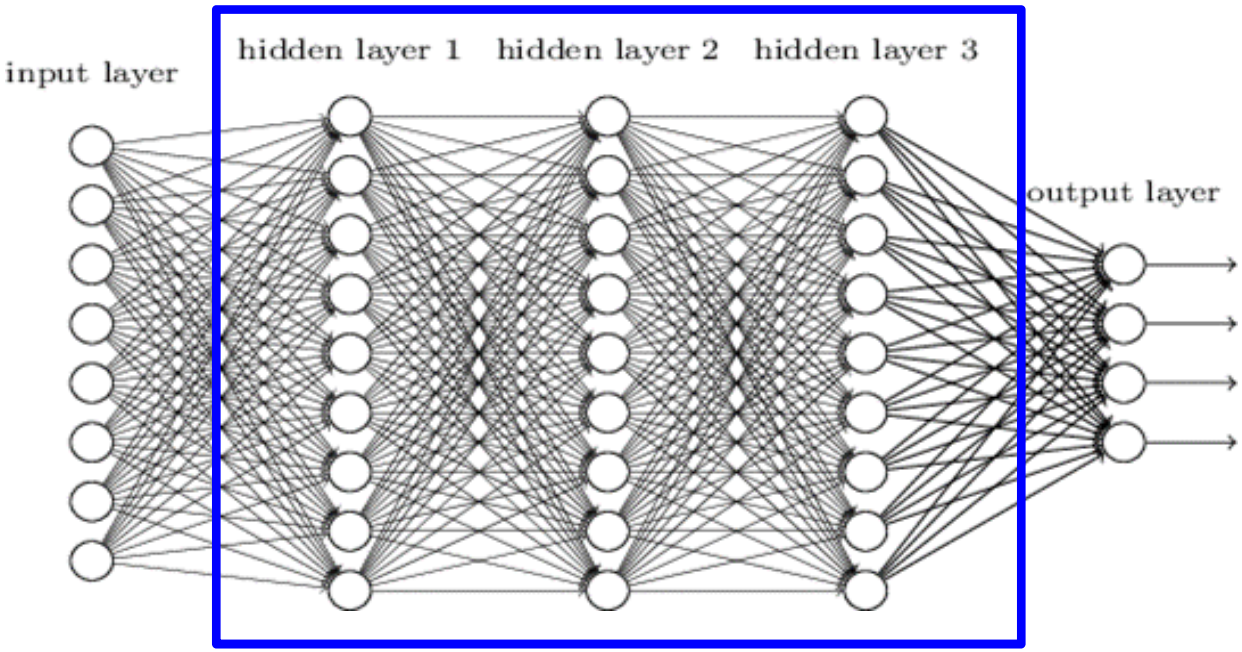
입력 : 16x16=256개 픽셀
(속성 개수 : 256개)

출력 : 26개(=알파벳 개수)의
확률값(0~1)

A : 0.9
B : 0.001
C : 0.002
D : 0.002
...
Z : 0.001

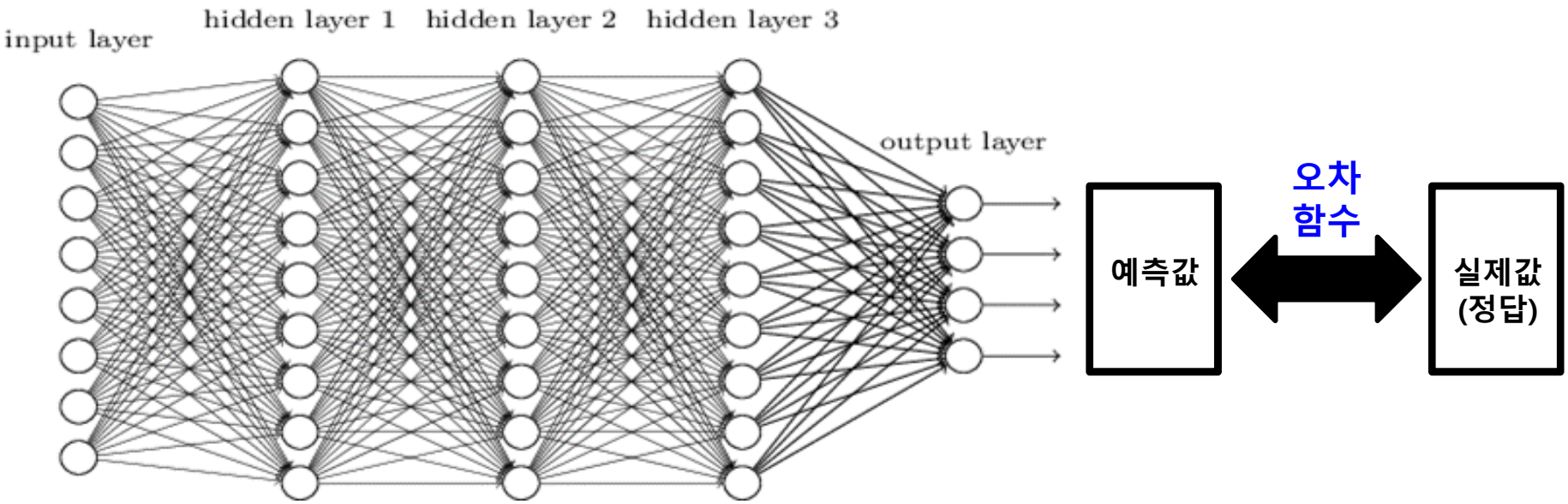
딥러닝 모델 구조 설계

- **은닉층 개수 설정**
  - 사용자가 임의로 설정 가능
    - ✓ 아래 그림에서는 층이 3개
- **은닉층 별 노드 개수 설정**
  - 사용자가 임의로 설정 가능
    - ✓ 아래 그림에서는 9개씩



딥러닝 모델 학습 설계

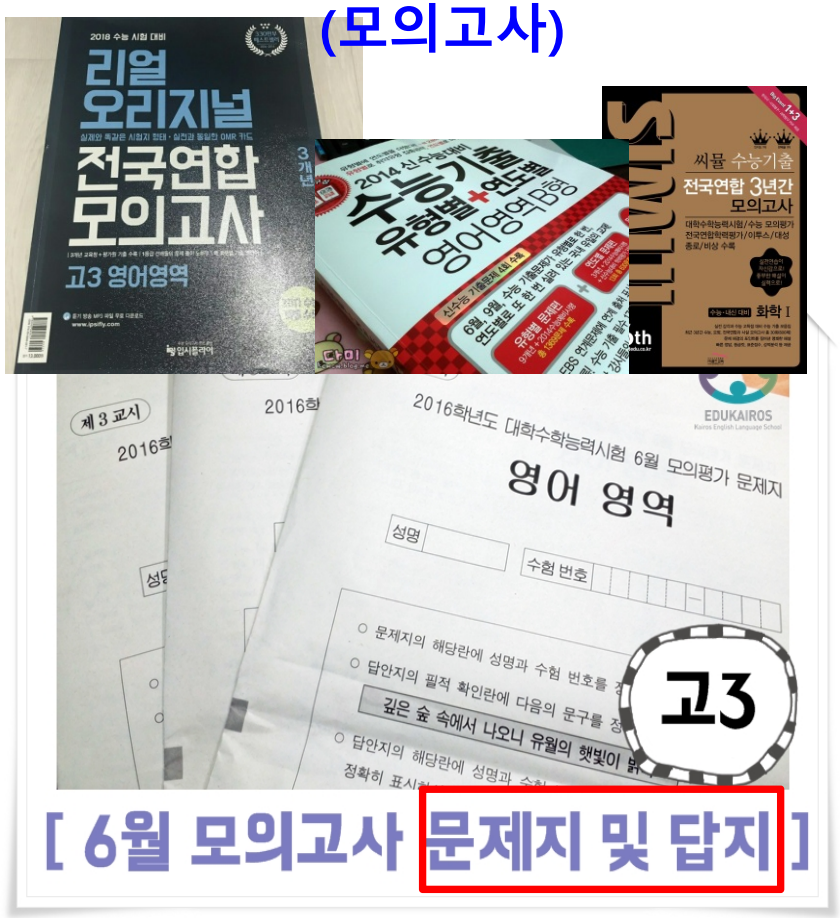
- 오차함수 설정
  - 대표적인 오차함수 : 평균 제곱근 오차
  - 그 밖에 : 평균 절대 오차, 이항 교차 엔트로피 등



딥러닝 모델 학습 설계

- 배치사이즈Batch size와 에포크Epoch 설정
  - 개념 이해

학습단계
(모의고사)



예측단계
(수능)

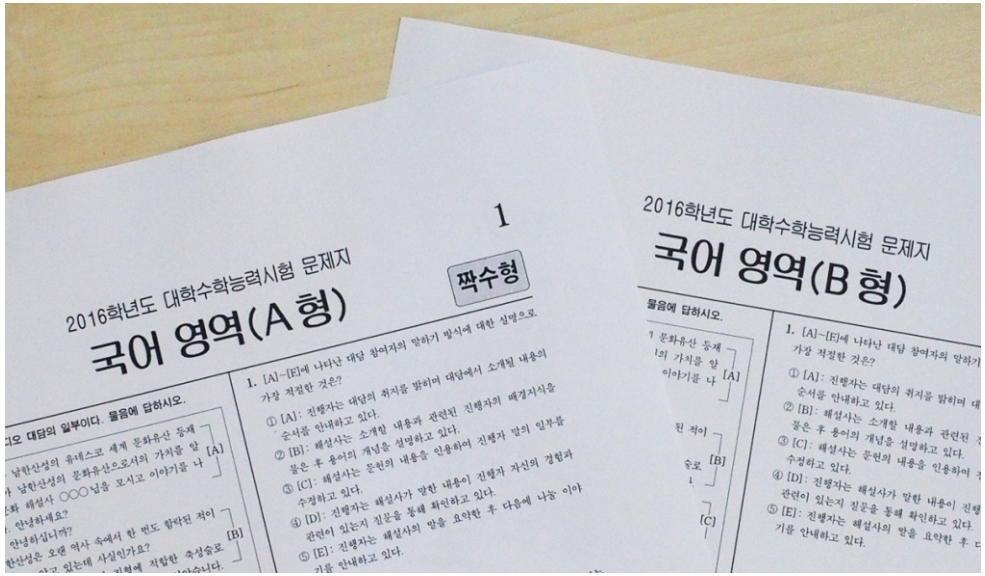


그림 출처 : 구글 검색

딥러닝 모델 학습 설계

■ 배치사이즈Batch size와 에포크Epoch 설정

- 개념 이해

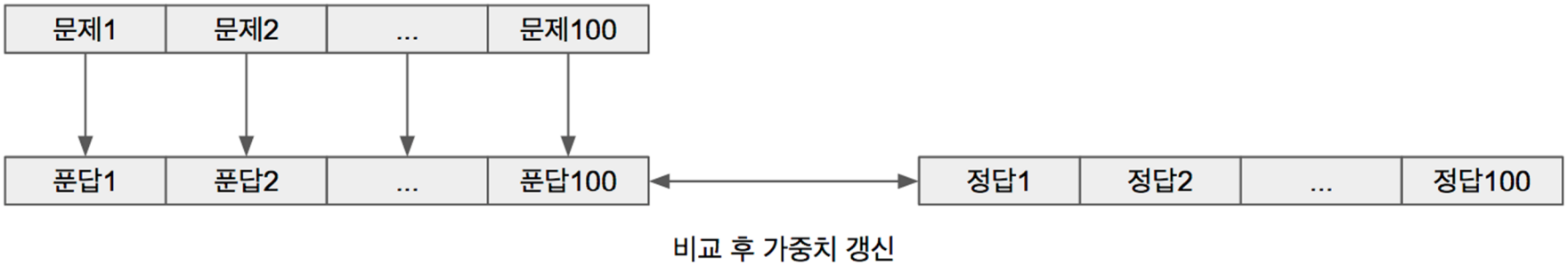
문제지	문제1	문제2	문제3	문제4	문제5	문제6	...	문제100
해답지	정답1	정답2	정답3	정답4	정답5	정답6	...	정답100

먼저 모의고사 1회분을 가지고 학습해봅시다. 이 1회분은 100문항이 있고, 해답지도 제공합니다. 문제를 푼 뒤 해답지와 맞춰보면서 학습이 이루어지기 때문에 해답지가 없으면 학습이 안 됩니다.

출처 :
https://tykimos.github.io/2017/03/25/Fit_Talk/

딥러닝 모델 학습 설계

- **배치사이즈Batch size와 에포크Epoch 설정**
  - **개념 이해 - 배치사이즈**

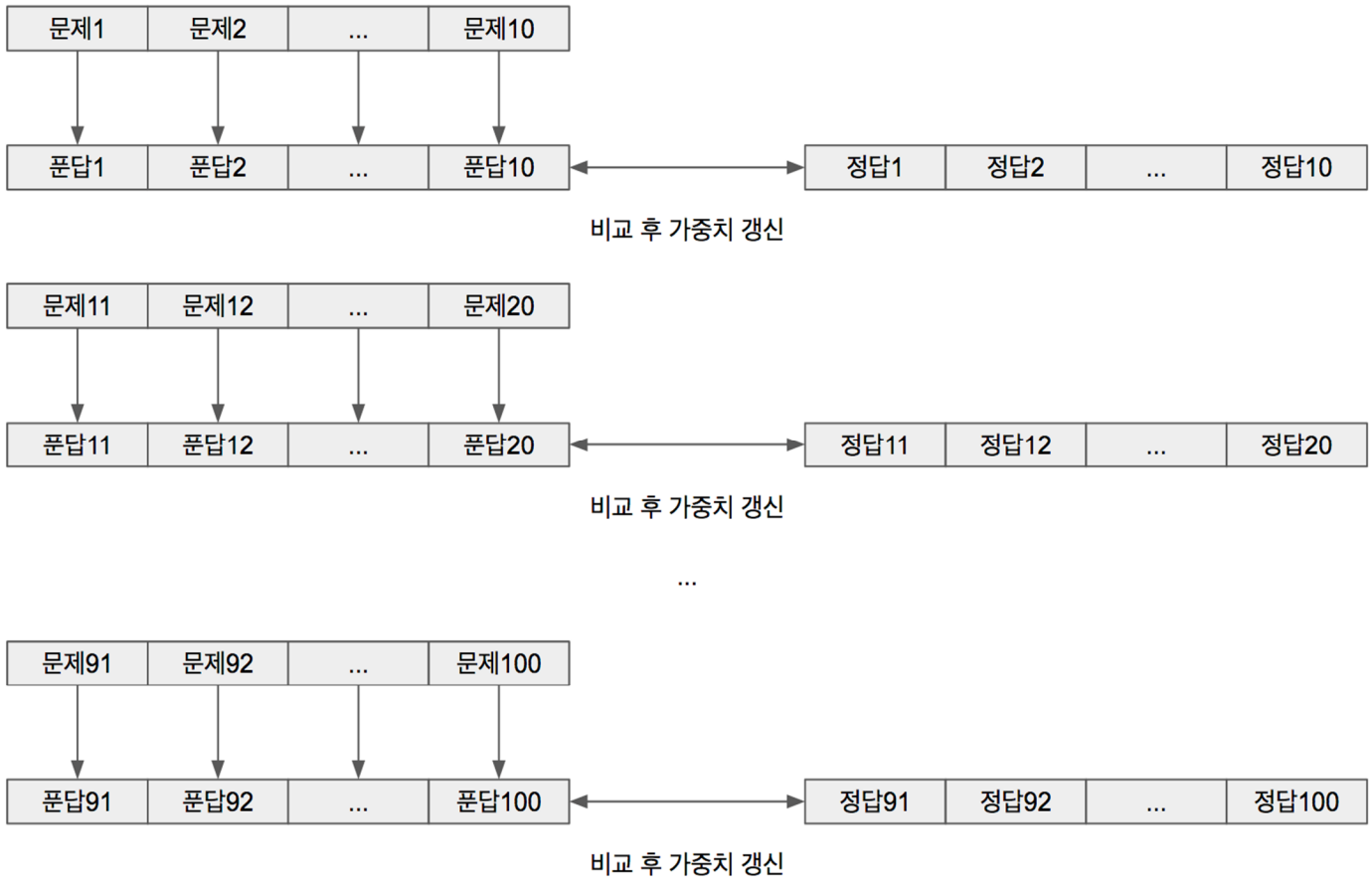


배치사이즈는 몇 문항을 풀고 해답을 맞추는 지를 의미합니다. 100문항일 때, 배치사이즈가 100이면 전체를 다 풀고 난 뒤에 해답을 맞춰보는 것입니다. 우리가 해답을 맞춰볼 때 '아하, 이렇게 푸는구나'라고 느끼면서 학습하는 것처럼 모델도 이러한 과정을 통해 가중치가 갱신됩니다. 전체 문제를 푼 뒤 해답과 맞추므로 이 때 가중치 갱신은 한 번만 일어납니다.

출처 :
https://tykimos.github.io/2017/03/25/Fit_Talk/

딥러닝 모델 학습 설계

- **배치사이즈Batch size와 에포크Epoch 설정**
  - **개념 이해 - 배치사이즈**



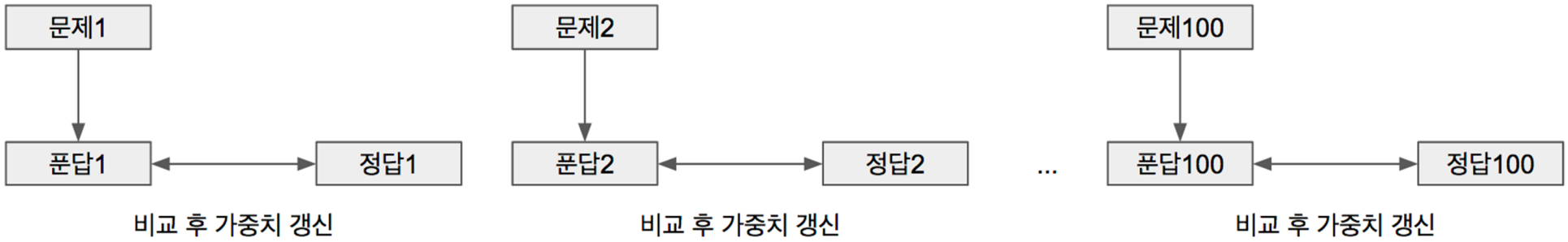
배치사이즈가 10이면 10문제씩 풀어보고 해답을 맞춰봅니다.

100문항을 10문제씩 나누어서 10번 해답을 맞추므로 가중치 갱신은 10번 일어납니다.

출처 :
https://tykimos.github.io/2017/03/25/Fit_Talk/

딥러닝 모델 학습 설계

- **배치사이즈Batch size와 에포크Epoch 설정**
  - **개념 이해 - 배치사이즈**



배치사이즈가 1이면 한 문제 풀고 해답 맞춰보고 또 한 문제 풀고 맞춰보고 하는 것입니다. 한 문제를 풀 때마다 가중치 갱신이 일어나므로 횟수는 100번입니다.

출처 :
https://tykimos.github.io/2017/03/25/Fit_Talk/

딥러닝 모델 학습 설계

- **배치사이즈Batch size와 에포크Epoch 설정**
  - **개념 이해 - 배치사이즈**

100문제 다 풀고 해답을 맞히는 것과 1문제씩 풀고 해답을 맞히는 것은 어떤 차이가 있을까?

모의고사 1회분에 비슷한 문항이 있다고 가정했을 때, 배치사이즈가 100일 때는 다 풀어보고 해답을 맞춰보기 때문에 한 문제를 틀릴 경우 이후 유사 문제를 모두 틀릴 가능성이 높음

배치사이즈가 1인 경우에는 한 문제씩 풀어보고 해답을 맞춰보기 때문에 유사문제 중 첫 문제를 틀렸다고 하더라도 해답을 보면서 학습하게 되므로 나머지 문제는 맞출 수도 있음

출처 :
https://tykimos.github.io/2017/03/25/Fit_Talk/

딥러닝 모델 학습 설계

- **배치사이즈Batch size와 에포크Epoch 설정**
  - **개념 이해 - 에포크**

에포크는 모의고사 1회분을 몇 번 풀어볼까 입니다. 즉 100문항의 문제들을 몇 번이나 반복해서 풀어보는 지 정하는 것입니다.

에포크가 20이면 모의고사 1회분을 20번 푸는 것입니다. 처음에는 같은 문제를 반복적으로 풀어보는 것이 무슨 효과가 있는 지 의문이 들었지만 우리가 같은 문제집을 여러 번 풀면서 점차 학습되듯이 모델도 같은 데이터셋으로 반복적으로 가중치를 갱신하면서 모델이 학습됩니다. 같은 문제라도 이전에 풀었을 때와 지금 풀었을 때가 학습상태(가중치)가 다르기 때문에 추가적인 학습이 일어납니다.

출처 :
https://tykimos.github.io/2017/03/25/Fit_Talk/

딥러닝 모델 학습 설계

- 배치사이즈Batch size와 에포크Epoch 설정
  - 개념 이해 - 에포크

출처 :
https://tykimos.github.io/2017/03/25/Fit_Talk/

코딩으로 딥러닝 모델 설계하기

- 필요한 라이브러리와 함수를 불러오기

```
# 딥러닝을 구동하는 데 필요한 케라스 함수를 불러옵니다.
from keras.models import Sequential
from keras.layers import Dense

# 필요한 라이브러리를 불러옵니다.
import numpy
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

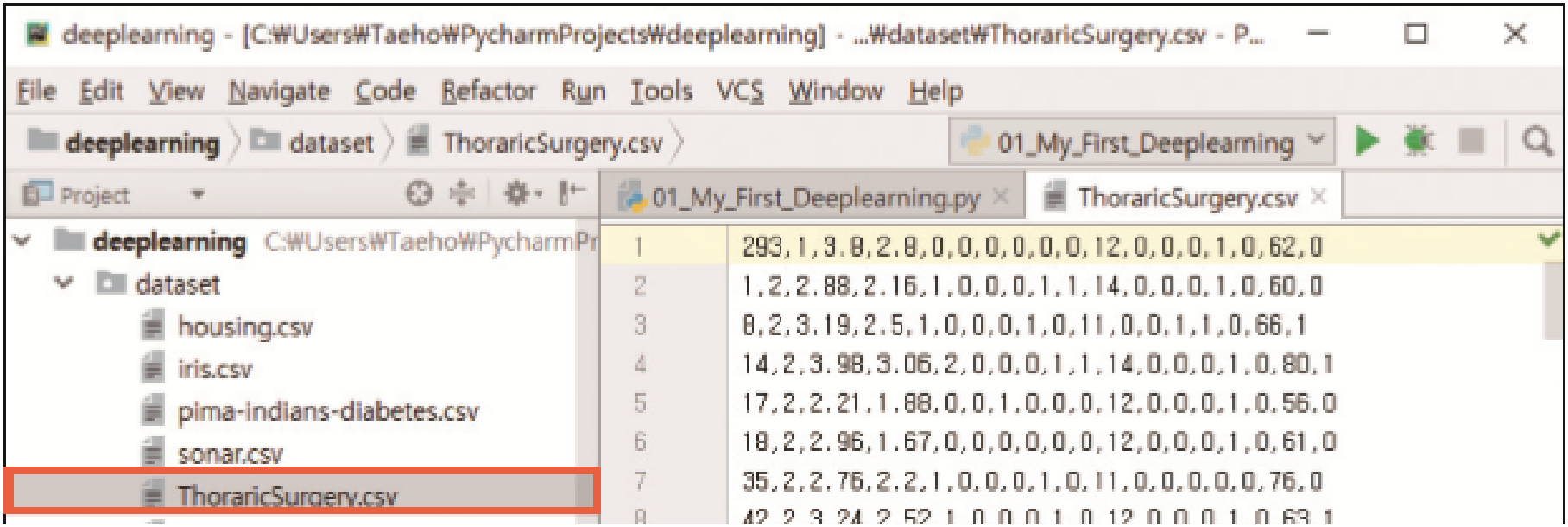
- ‘ThoracicSurgery.csv’ 파일 : 폴란드의 브로츠와프 의과대학에서 2013년 공개한 폐암 수술 환자의 수술 전 진단 데이터와 수술 후 생존 결과를 기록한 실제 의료 기록 데이터

출처 :
https://archive.ics.uci.edu/ml/datasets/Thoracic+Surgery+Data

```
# 준비된 수술 환자 데이터를 불러들입니다.
Data_set = numpy.loadtxt("ThoracicSurgery.csv", delimiter=",")
```

코딩으로 딥러닝 모델 설계하기

- 'ThoracicSurgery.csv' 파일 들여다 보기
  - ✓ 모두 470라인으로 이루어져 있고, 각 라인은 18개 항목으로 구분



KOREATECH 한국기술교육대학교

출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

- 모두 470라인으로 이루어져 있고, 각 라인은 18개 항목으로 구분
  - ✓ 1~17번 항목 : 속성 (종양 유형, 폐활량, 호흡 곤란 여부, 고통 정도, 기침, 흡연, 천식 여부 등)
  - ✓ 18번 : 클래스 (1: 사망, 0: 생존)

속성 / 클래스

	정보 1	정보 2	정보 3	…	정보 17	생존 여부
1번째 환자	293	1	3.8	…	62	0
2번째 환자	1	2	2.88	…	60	0
3번째 환자	8	3	3.19	…	66	1
…	…	…	…	…	…	…
470번째 환자	447	8	5.2	…	49	0

샘플

KOREATECH 한국기술교육대학교

출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

- 환자의 기록과 생존 여부를 X와 Y로 구분하여 저장

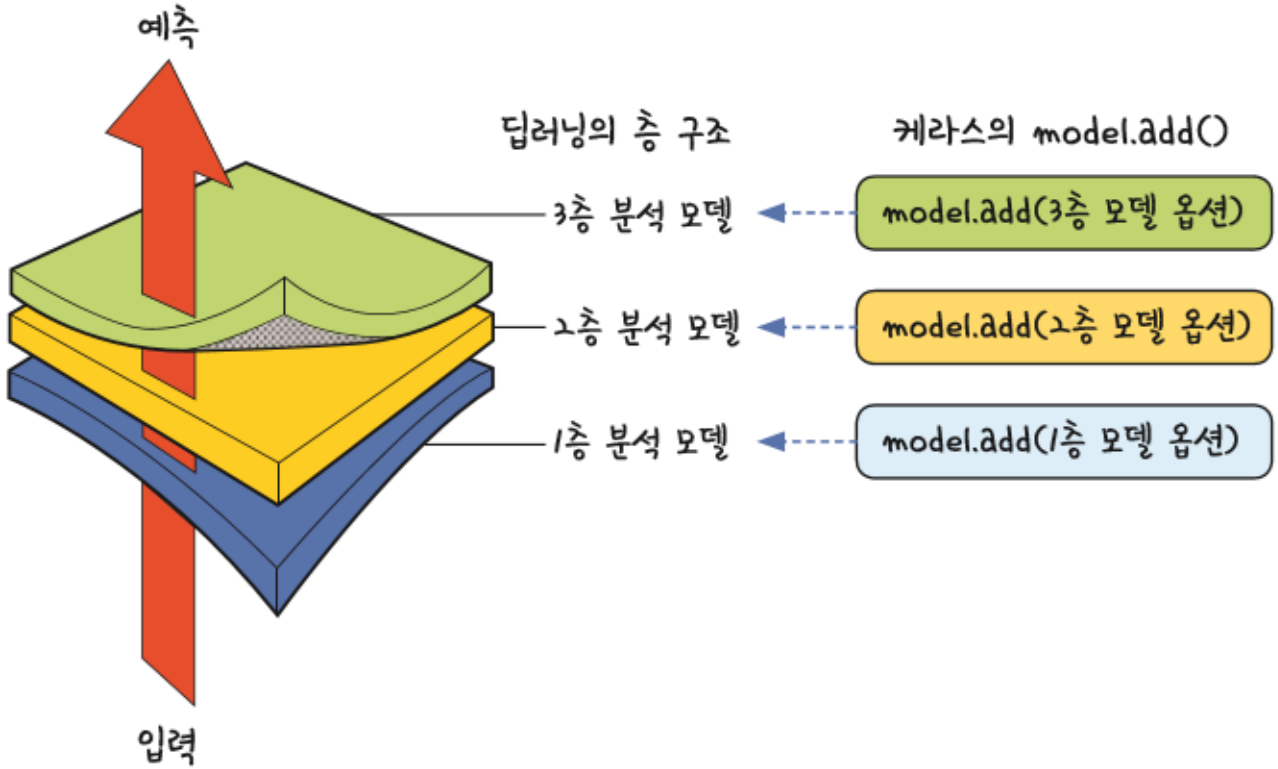
```
X = Data_set[:,0:17] # 속성
Y = Data_set[:,17] # 클래스
```

속성 | 클래스

항목 \ 줄	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	293	1	3.8	2.8	0	0	0	0	0	0	12	0	0	0	1	0	62	0
2	1	2	2.88	2.16	1	0	0	0	1	1	14	0	0	0	1	0	60	0
3	8	2	3.19	2.5	1	0	0	0	1	0	11	0	0	1	1	0	66	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
470	447	8	5.2	4.1	0	0	0	0	0	0	12	0	0	0	0	0	49	0

출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기



출처 : 모두의 딥러닝(조태호)

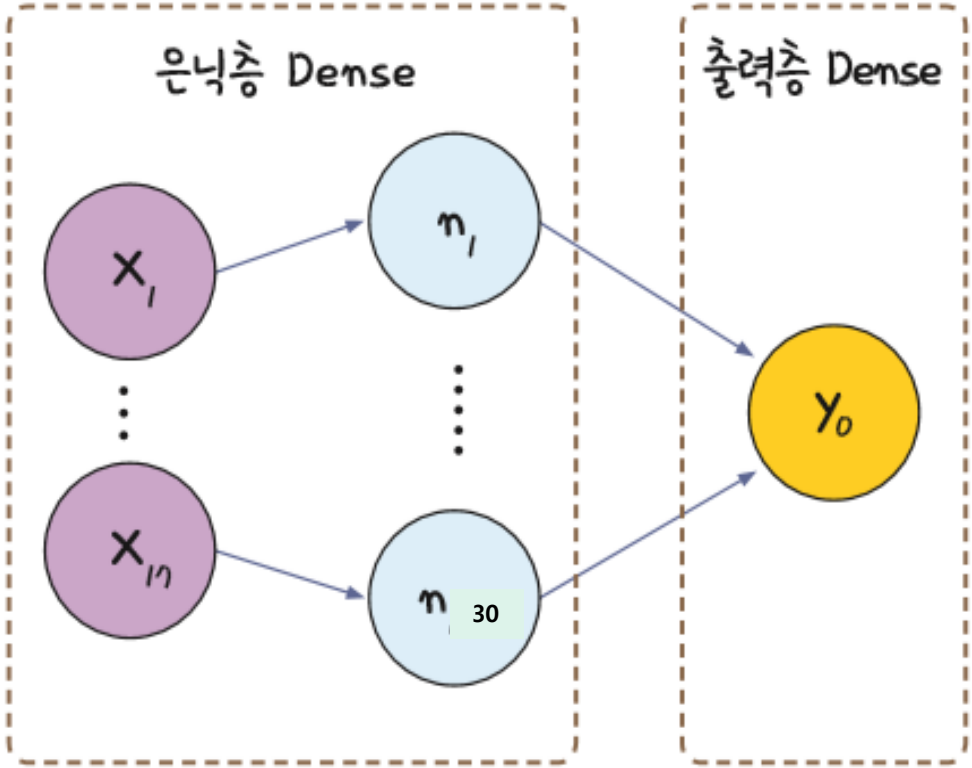
코딩으로 딥러닝 모델 설계하기

- 딥러닝 구조 설계 : 퍼셉트론 층을 차곡차곡 추가
  - ✓ Sequential() 함수로 모델 선언
  - ✓ model.add()라는 라인을 추가하면 새로운 층이 만들어짐
  - ✓ 각 층은 Dense()라는 함수를 통해 구체적으로 구조가 결정됨

```
model = Sequential()      # 모델 선언
model.add(Dense(30, input_dim=17, activation='relu'))  # 입력층과 은닉층 설계
model.add(Dense(1, activation='sigmoid'))          # 출력층 설계
```


출처 : 모두의 딥러닝(조태호)

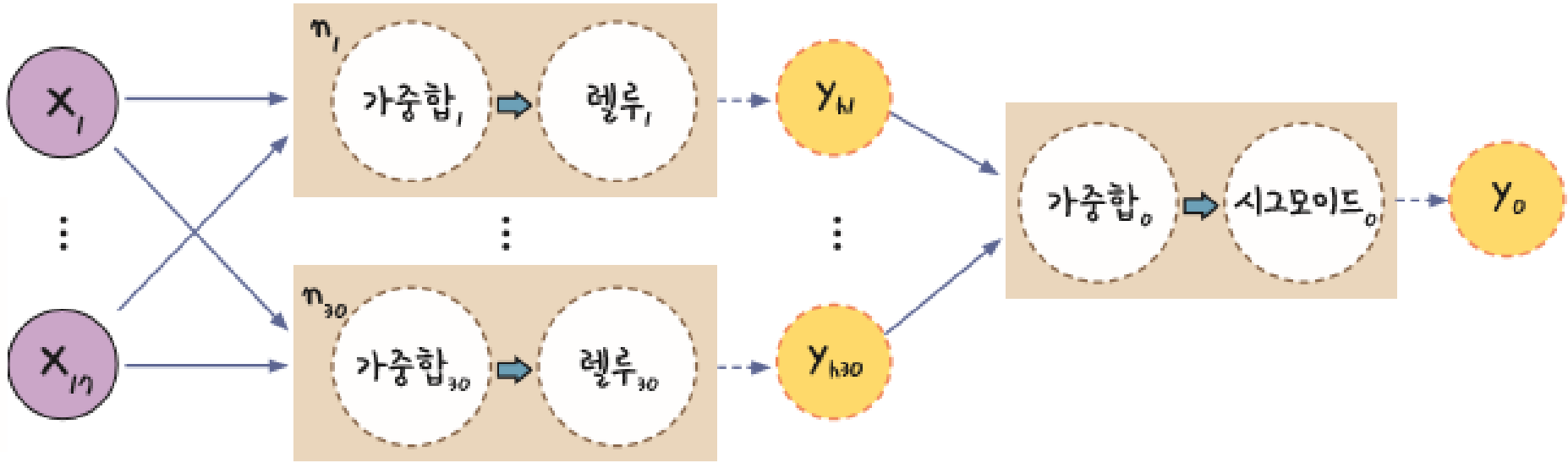
코딩으로 딥러닝 모델 설계하기




```
model = Sequential()      # 모델 선언
model.add(Dense(30, input_dim=17, activation='relu'))   # 입력층과 은닉층 설계
model.add(Dense(1, activation='sigmoid'))          # 출력층 설계
```


출처 : 모두의 딥러닝(조태호)

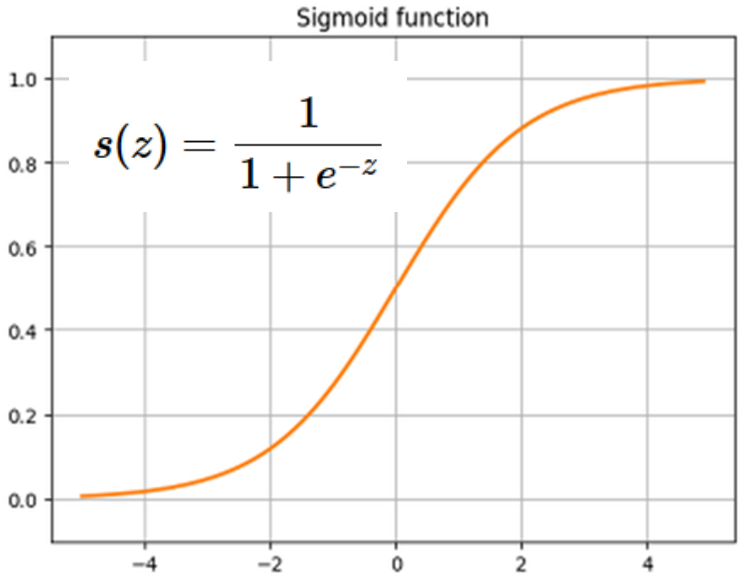
코딩으로 딥러닝 모델 설계하기



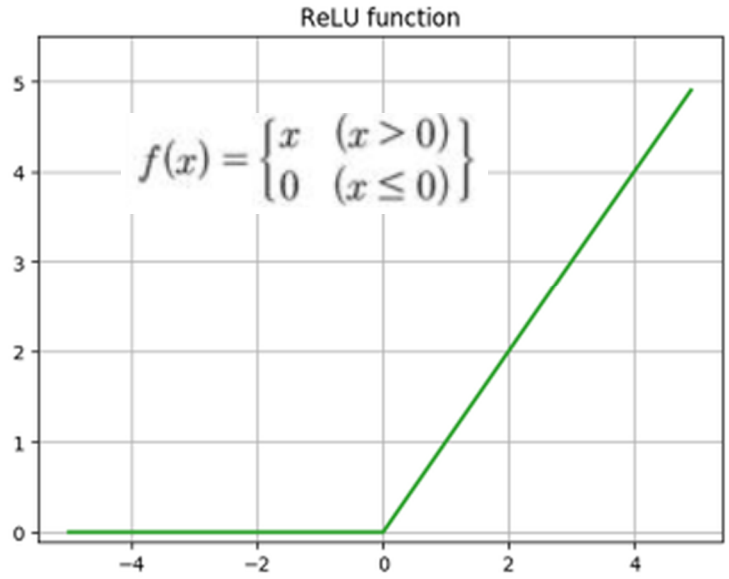

```
model = Sequential()  # 모델 선언
model.add(Dense(30, input_dim=17, activation='relu'))
# 입력층과 은닉층 설계 : 데이터에서 17개의 값을 받아 은닉층의 30개 노드로 보낸다는 뜻
# 렐루(relu) 함수를 활성화 함수로 사용
model.add(Dense(1, activation='sigmoid'))
# 출력층 설계 : 출력층의 노드 수는 1개, 시그모이드 함수를 활성화 함수로 사용
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기



시그모이드 함수



렐루(ReLU) 함수

```
model = Sequential()  # 모델 선언
model.add(Dense(30, input_dim=17, activation='relu'))
# 입력층과 은닉층 설계 : 데이터에서 17개의 값을 받아 은닉층의 30개 노드로 보낸다는 뜻
# 렐루(relu) 함수를 활성화 함수로 사용
model.add(Dense(1, activation='sigmoid'))
# 출력층 설계 : 출력층의 노드 수는 1개, 시그모이드 함수를 활성화 함수로 사용
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

- model.compile() : 앞서 지정한 모델에 대해 구체적인 학습 조건 설정

  ✓ loss 옵션 : 신경망의 오차함수(=손실함수)의 종류를 설정
  - 여기서는 평균 제곱근 오차 사용

  ✓ optimizer 옵션 : 설정한 오차함수의 값을 줄여나가는 최적화 알고리즘의 종류
  - sgd : stochastic gradient descent (기본적인 경사 하강법)

  ✓ metric 옵션 : 모델의 성능을 평가하는 척도 설정
  - 여기서는 정확도Accuracy를 성능 평가 척도로 사용

```
model.compile(loss='mean_squared_error', optimizer='sgd', metrics=['accuracy'])
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

- [참고] 케라스에서 사용되는 대표적인 오차함수

* 실제 값을 yt, 예측 값을 yo라고 가정할 때

평균 제곱 계열	mean_squared_error	평균 제곱 오차 계산: mean(square(yt − yo))
	mean_absolute_error	평균 절대 오차(실제 값과 예측 값 차이의 절댓값 평균) 계산: mean(abs(yt − yo))
	mean_absolute_percentage_error	평균 절대 백분율 오차(절댓값 오차를 절댓값으로 나눈 후 평균) 계산: mean(abs(yt − yo)/abs(yt) (단, 분모 ≠ 0)
	mean_squared_logarithmic_error	평균 제곱 로그 오차(실제 값과 예측 값에 로그를 적용한 값의 차이를 제곱한 값의 평균) 계산: mean(square((log(yo) + 1) − (log(yt) + 1)))
교차 엔트로피 계열	categorical_crossentropy	범주형 교차 엔트로피(일반적인 분류)
	binary_crossentropy	이항 교차 엔트로피(두 개의 클래스 중에서 예측할 때)

```
model.compile(loss='mean_squared_error', optimizer='sgd', metrics=['accuracy'])
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

- model.fit() : 모델을 실행하는 함수

  ✓ epoch=30 : 모든 샘플이 30번 재사용될 때까지 실행을 반복하라는 뜻
  (모든 샘플 470개 x 30번 재사용)

  ✓ batch_size=10 : 모든 샘플 470개를 10개씩 끊어서 넣으라는 뜻

```
model.fit(X, Y, epochs=30, batch_size=10)
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

- 딥러닝 모델 평가 결과를 출력

```
print("\n Accuracy: %.4f" % (model.evaluate(X, Y)[1]))
```

In: model.evaluate(X, Y)
Out: [0.14480824178837715, 0.8510638285190502]

학습된 딥러닝 모델의 손실(오차함수 값)

학습된 딥러닝 모델의 정확도

출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

- 최종 소스코드

```
# 딥러닝을 구동하는 데 필요한 케라스 함수를 불러옵니다.
from keras.models import Sequential
from keras.layers import Dense

# 필요한 라이브러리를 불러옵니다.
import numpy
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

```
# 준비된 수술 환자 데이터를 불러들입니다.
Data_set = numpy.loadtxt("ThoracicSurgery.csv", delimiter=",")

# 환자의 기록과 수술 결과를 X와 Y로 구분하여 저장합니다.
X = Data_set[:,0:17]
Y = Data_set[:,17]

# 딥러닝 구조를 결정합니다(모델을 설정하고 실행하는 부분입니다).
model = Sequential()
model.add(Dense(30, input_dim=17, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```


출처 : 모두의 딥러닝(조태호)

코딩으로 딥러닝 모델 설계하기

```
# 딥러닝을 실행합니다.
model.compile(loss='mean_squared_error', optimizer='sgd', metrics=['accuracy'])
model.fit(X, Y, epochs=30, batch_size=10)

# 결과를 출력합니다.
print("\n Accuracy: %.4f" % (model.evaluate(X, Y)[1]))
```

- 실행 결과
  - ✓ 85.11% : 예측 정확도
  - ✓ 기존 폐암 환자의 데이터를 딥러닝에 넣고 학습시킨 결과, 새로운 환자의 수술 후 생존을 100번 중 약 85번은 맞힐 수 있다는 것을 의미

출처 : 모두의 딥러닝(조태호)