AI 라이브러리 활용

10장 모델 설계하기

이 찬 우

학습내용 : 모델 설계하기

1 | 모델의 정의

2 | 입력층, 은닉층, 출력층

3 | 모델 컴파일

4 | 교차 엔트로피

5 | 모델 실행하기

모델 설계하기

- **실습 데이터** 폐암 수술 환자의 생존율 예측
  - dataset/ThoracicSurgery.csv
- 딥러닝의 기본 개념들이 실전에서는 어떤 방식으로 구현되는지, 왜 우리가 그 어려운 개념들을 익혀야 했는지를 공부하겠음

1 | 모델의 정의

코드 10-1 폐암 수술 환자의 생존율 예측하기 ②

- 예제 소스 run_project/01_My_First_Deeplearning.ipynb

```
# 딥러닝을 구동하는 데 필요한 케라스 함수 호출
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# 필요한 라이브러리 불러옴
import numpy as np
import tensorflow as tf

# 실행할 때마다 같은 결과를 출력하기 위해 설정하는 부분
np.random.seed(3)
tf.random.set_seed(3)
```

1 | 모델의 정의

```
# 준비된 수술 환자 데이터를 불러옴
Data_set = np.loadtxt("../dataset/ThoracicSurgery.csv",
delimiter=",")

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

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

1 | 모델의 정의

```
# 딥러닝 실행
model.compile(loss='mean_squared_error', optimizer='adam',
metrics=['accuracy'])
model.fit(X, Y, epochs=100, batch_size=10)
```

1 | 모델의 정의

- 딥러닝의 모델을 설정하고 구동하는 부분은 모두 model 이라는 함수를 선언하며 시작이 됨
- 먼저 model = Sequential( )로 시작되는 부분은 딥러닝의 구조를 짜고 층을 설정하는 부분임
- 이어서 나오는 model.compile( ) 부분은 위에서 정해진 모델을 컴퓨터가 알아들을 수 있게끔 컴파일 하는 부분임
- model.fit( )으로 시작하는 부분은 모델을 실제로 수행하는 부분임

2 | 입력층, 은닉층, 출력층

- 먼저 딥러닝의 구조를 짜고 층을 설정하는 부분을 살펴보면 다음과 같음

```
model = Sequential( )
model.add(Dense(30, input_dim=17, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

- Sequential( ) 함수를 model로 선언해 놓고 model.add( )라는 라인을 추가하면 새로운 층이 만들어짐
- 코드에는 model.add( )로 시작되는 라인이 두 개가 있으므로 두 개의 층을 가진 모델을 만든 것임

2 | 입력층, 은닉층, 출력층

- 맨 마지막 층은 결과를 출력하는 '출력층'이 됨
- 나머지는 모두 '은닉층'의 역할을 함
- 지금 만들어진 이 두 개의 층은 각각 은닉층과 출력층임

2 | 입력층, 은닉층, 출력층

- 각각의 층은 Dense라는 함수를 통해 구체적으로 그 구조가 결정됨
- Dense(30, input_dim=17, activation='relu')부분을 더 살펴보자

  → model.add( ) 함수를 통해 새로운 층을 만들고 나면

  Dense() 함수를 통해 이 층에 몇 개의 노드를 만들 것인지를 숫자로 써줌

  → 30이라고 되어 있는 것은 이 층에 30개의 노드를 만들겠다는 것

  → 이어서 input_dim이라는 변수가 나옴

  → 이는 입력 데이터에서 몇 개의 값을 가져올지를 정하는 것

2 | 입력층, 은닉층, 출력층

- keras는 입력층을 따로 만드는 것이 아니라,

  첫 번째 은닉층에 input_dim을 적어줌으로써

  첫 번째 Dense가 은닉층 + 입력층의 역할을 겸함
- 우리가 다루고 있는 폐암 수술 환자의 생존 여부 데이터에는 17개의 입력 값들이 있음
- 데이터에서 17개의 값을 받아 은닉층의 30개 노드로 보낸다는 뜻

2 | 입력층, 은닉층, 출력층

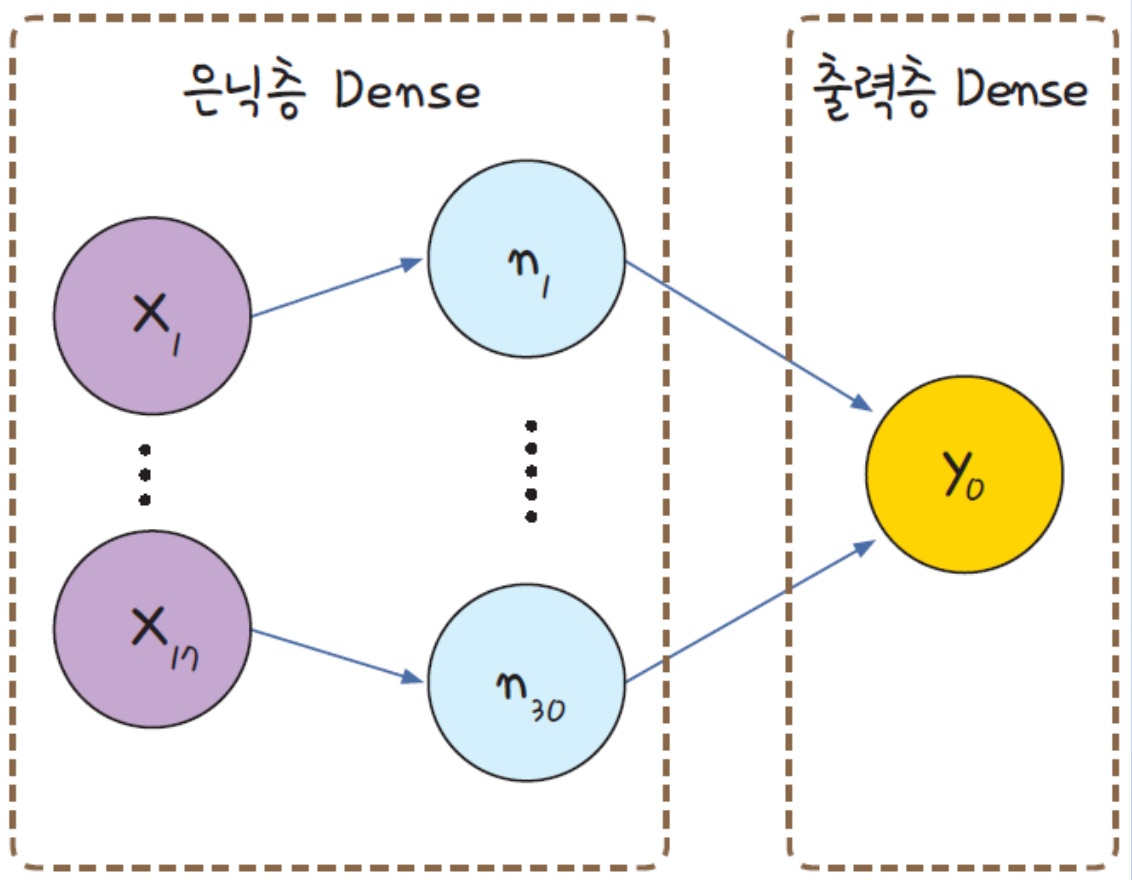

그림 10-1 첫 번째 Dense는 입력층과 첫 번째 은닉층을, 두 번째 Dense는 출력층을 의미

2 | 입력층, 은닉층, 출력층

- 은닉층의 각 노드는 17개의 입력 값에서 임의의 가중치를 가지고
  각 노드로 전송되어 활성화 함수를 만남
- 활성화 함수를 거친 결괏값이 출력층으로 전달됨
- 다음에 나오는 activation 부분에 우리가 원하는 활성화 함수를 적어 주면 됨
- 여기서는 앞서 배운 렐루를 사용함

2 | 입력층, 은닉층, 출력층

- 이제 두 번째 나오는 model.add(Dense(1, activation='sigmoid'))를 보겠음
- 마지막 층이므로 이 층이 곧 출력층이 됨
- 출력 값을 하나로 정해서 보여 줘야 하므로 출력층의 노드 수는 1개임
- 이 노드에서 입력받은 값 역시 활성화 함수를 거쳐 최종 출력 값으로 나와야 함
- 여기서는 시그모이드(sigmoid)를 활성화 함수로 사용함

3 | 모델 컴파일

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

- model.compile 부분은 앞서 지정한 모델이 효과적으로 구현될 수 있게 여러 가지 환경을 설정해 주면서 컴파일하는 부분임
- 먼저 어떤 오차 함수를 사용할지를 정해야 함
- 여기서는 평균 제곱 오차 함수(mean_squared_error)를 사용함

3 | 모델 컴파일

- 오차 함수에는 평균 제곱 오차 계열의 함수 외에도 교차 엔트로피 계열의 함수가 있음
- 평균 제곱 오차 계열의 함수는 수렴하기까지 속도가 많이 걸린다는 단점이 있음
- 교차 엔트로피 계열의 함수는 출력 값에 로그를 취해서, 오차가 커지면 수렴 속도가 빨라지고 오차가 작아지면 수렴 속도가 감소하게끔 만든 것

3 | 모델 컴파일

- 최적화를 위한 방법도 간단히 고를 수 있음

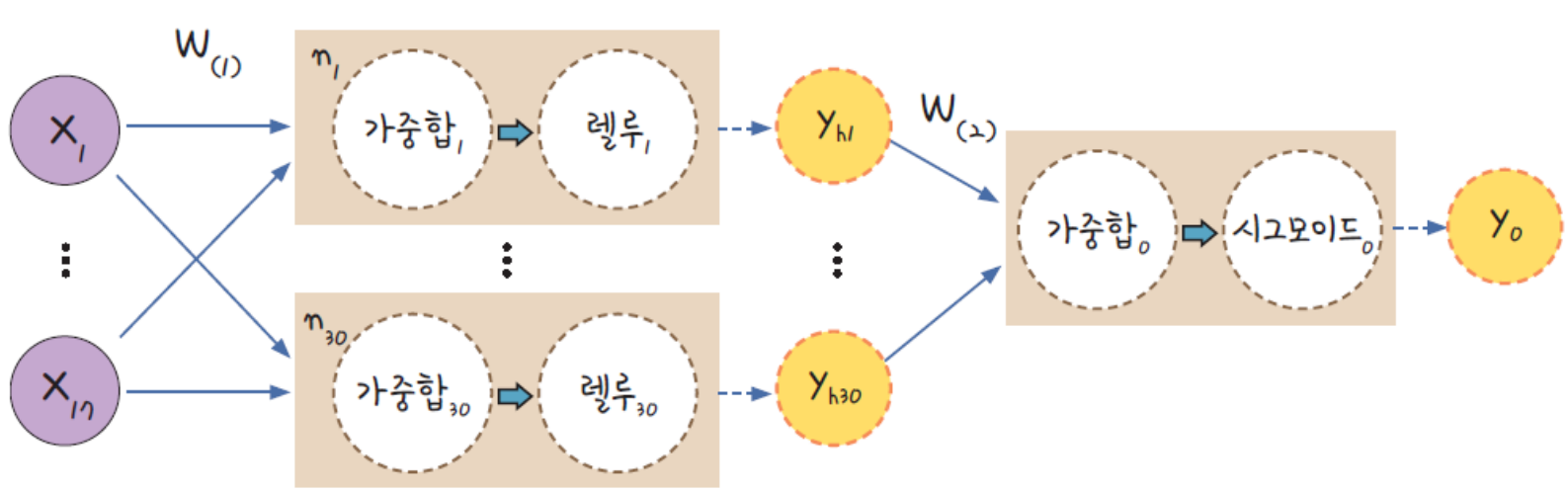

그림 10-2 폐암 환자 생존율 예측 신경망 모델의 도식화. 여기서 W는 각 층별 가중치(w)들의 집합을 말함.

3 | 모델 컴파일

- metrics() 함수는 모델이 컴파일 될 때 모델 수행 결과를 나타내게끔 설정하는 부분
- 정확도를 측정하기 위해 사용되는 테스트 샘플을 학습 과정에서 제외시킴으로써 과적합 문제를 방지하는 기능을 담고 있음

4 | 교차 엔트로피

- 교차 엔트로피는 주로 분류 문제에서 많이 사용되는데,
  특별히 예측 값이 참과 거짓 둘 중 하나인 형식일 때는
  binary_crossentropy(이항 교차 엔트로피)를 씀
- binary_crossentropy를 적용하려면 앞 식의
  mean_squared_error 자리에 binary_crossentropy를 입력하면 됨
- 이를 실행하면 예측 정확도(Accuracy)가 약간 향상되는 것을 알 수 있음

4 | 교차 엔트로피

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)

표 10-1 대표적인 오차 함수.

* 실제 값을 yt, 예측 값을 yo라고 가정할 때

5 | 모델 실행하기

- 컴파일 단계에서 정해진 환경을 주어진 데이터를 불러 실행시킬 때 사용되는 함수는 다음과 같이 model.fit( ) 부분임

```
model.fit(X, Y, epochs=100, batch_size=10)
```

5 | 모델 실행하기

- 속성 :

  주어진 폐암 수술 환자의 생존 여부 데이터는

  총 470명의 환자에게서 17개의 정보를 정리한 것이고 이때 각 정보를 말함

- 샘플 :

  생존 여부를 클래스, 가로 한 줄에 해당하는 각 환자의 정보

  주어진 데이터에는 총 470개의 샘플이 각각 17개씩의 속성을 가지고 있는 것

5 | 모델 실행하기

속성 / 클래스

	정보 1	정보 2	정보 3	…	정보 17	생존 여부
샘플 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

표 10-2 폐암 환자 생존율 예측 데이터의 샘플, 속성, 클래스 구분

TIP

이 용어는 출처마다 조금씩 다릅니다. 예를 들어 샘플을 instance 또는 example이라 부르기도 하고, 속성 대신 피처(feature)라고 부르기도 합니다. 이 책에서는 '속성'과 '샘플'로 통일하여 사용하겠습니다.

5 | 모델 실행하기

- 1 epoch('에포크'라고 읽음) :

  학습 프로세스가 모든 샘플에 대해 한 번 실행되는 것
- 코드에서 epochs=100으로 지정한 것은 각 샘플이

  처음부터 끝까지 100번 재사용될 때까지 실행을 반복하라는 뜻

5 | 모델 실행하기

- batch_size
  - 샘플을 한 번에 몇 개씩 처리할지를 정하는 부분으로

    batch_size=10은 전체 470개의 샘플을 10개씩 끊어서 집어넣으라는 뜻
  - batch_size가 너무 크면 학습 속도가 느려지고,

    너무 작으면 각 실행 값의 편차가 생겨서 전체 결괏값이 불안정해질 수 있음
  - 자신의 컴퓨터 메모리가 감당할 만큼의 batch_size를 찾아 설정해 주는 것이 좋음

정리 학습 : 모델 설계하기

1 | 모델의 정의

2 | 입력층, 은닉층, 출력층

3 | 모델 컴파일

4 | 교차 엔트로피

5 | 모델 실행하기

다음 수업

파이썬의 객체와 클래스 부분을 학습