강사 윤영주
yj.youn1103@gmail.com

둘째마당
예측 모델의 기본 원리

6장 로지스틱 회귀 모델: 참 거짓 판단하기

1 로지스틱 회귀의 정의

2 시그모이드 함수

3 오차 공식

4 로그 함수

5 텐서플로에서 실행하는 로지스틱 회귀 모델

로지스틱 회귀 모델: 참 거짓 판단하기

- **로지스틱 회귀 모델: 참 거짓 판단하기**
  - 법정 드라마나 영화를 보면 검사가 피고인을 다그치는 장면이 종종 나옴
  - 검사의 예리한 질문에 피고인이 당황한 표정으로 변명을 늘어놓을 때 검사가 이렇게 소리침
  - “예, 아니요로만 대답하세요!”
  - 때로 할 말이 많아도 ‘예’ 혹은 ‘아니요’로만 대답해야 할 때가 있음
  - 실은 이와 같은 상황이 딥러닝에서도 끊임없이 일어남
  - 전달받은 정보를 놓고 참과 거짓 중 하나를 판단해 다음 단계로 넘기는 장치들이 딥러닝 내부에서 쉬지 않고 작동하는 것
  - 딥러닝을 수행한다는 것은 겉으로 드러나지 않는 ‘미니 판단 장치’들을 이용해서 복잡한 연산을 해낸 끝에 최적의 예측 값을 내놓는 작업이라고 할 수 있음

로지스틱 회귀 모델: 참 거짓 판단하기

- **로지스틱 회귀 모델: 참 거짓 판단하기**

- 참과 거짓 중 하나를 내놓는 과정은 **로지스틱 회귀**(logistic regression)의 원리를 거쳐 이루어짐
- 이제 회귀 분석의 또 다른 토대를 이루는 로지스틱 회귀에 대해 알아보자

1 로지스틱 회귀의 정의

1 로지스틱 회귀의 정의

- **로지스틱 회귀의 정의**
  - 5장에서 공부한 시간과 성적 사이의 관계를 좌표에 나타냈을 때, 좌표의 형태가 직선으로 해결되는 선형 회귀를 사용하기에 적절했음을 보았음
  - 직선으로 해결하기에 적절하지 않은 경우도 있음
  - 점수가 아니라 오직 합격과 불합격만 발표되는 시험이 있다고 하자
  - 공부한 시간에 따른 합격 여부를 조사해 보니 표 6-1과 같았음

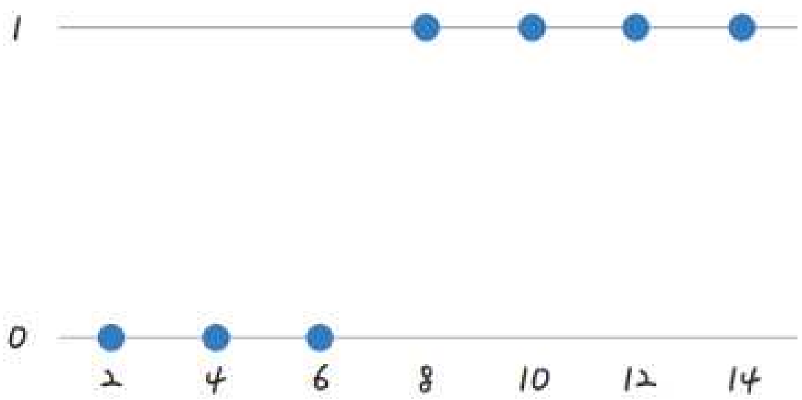
▼ **표 6-1 | 공부한 시간에 따른 합격 여부**

공부한 시간	2	4	6	8	10	12	14
합격 여부	불합격	불합격	불합격	합격	합격	합격	합격

1 로지스틱 회귀의 정의

- **로지스틱 회귀의 정의**
  - 합격을 1, 불합격을 0이라고 하고, 이를 좌표 평면에 표현하면 그림 6-1과 같음

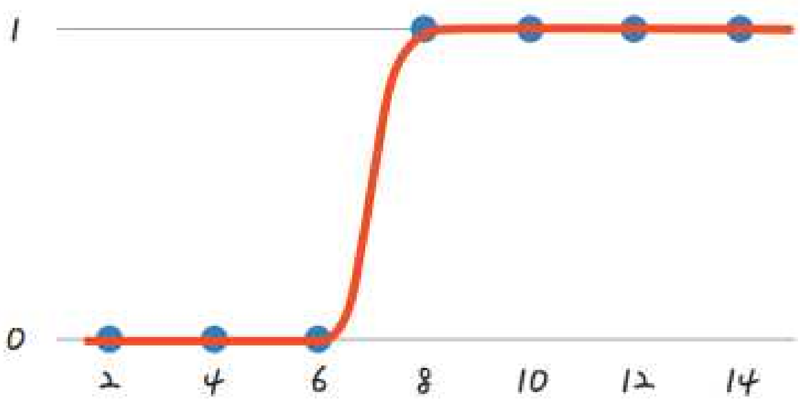
▼ **그림 6-1 | 합격과 불합격만 있을 때의 좌표 표현**



1 로지스틱 회귀의 정의

- **로지스틱 회귀의 정의**
  - 앞 장에서 배운 대로 선을 그어 이 점의 특성을 잘 나타내는 일차 방정식을 만들 수 있을까?
  - 이 점들은 1과 0 사이의 값이 없으므로 직선으로 그리기가 어려움
  - 점들의 특성을 정확하게 담아내려면 직선이 아니라 다음과 같이 S자 형태여야 함

▼ **그림 6-2 | 각 점의 특성을 담은 선을 그었을 때**



1 로지스틱 회귀의 정의

- **로지스틱 회귀의 정의**
  - 로지스틱 회귀는 선형 회귀와 마찬가지로 적절한 선을 그려 가는 과정
  - 다만, 직선이 아니라 참(1)과 거짓(0) 사이를 구분하는 S자 형태의 선을 그어 주는 작업

2 시그모이드 함수

2 시그모이드 함수

- **시그모이드 함수**
  - 이러한 S자 형태로 그래프가 그려지는 함수가 있음
  - 바로 우리가 '3장. 딥러닝을 위한 기초 수학'에서 배운 시그모이드 함수(sigmoid function)
  - 시그모이드 함수를 이용해 로지스틱 회귀를 풀어 나가는 공식은 다음과 같음

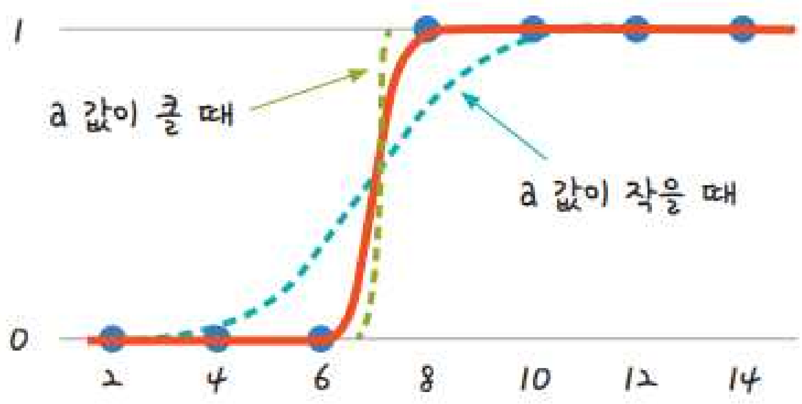
$$y = \frac{1}{1 + e^{-(ax+b)}}$$

  - 이 식을 통해 알 수 있는 것은 우리가 구해야 하는 값이 여기서도 결국 ax + b라는 것

2 시그모이드 함수

- **시그모이드 함수**
  - 선형 회귀 때와 마찬가지임
  - 여기서 a와 b는 무슨 의미를 가지고 있을까?
  - 먼저 a는 그래프의 경사도를 결정
  - 그림 6-3과 같이 a 값이 커지면 경사가 커지고 a 값이 작아지면 경사가 작아짐

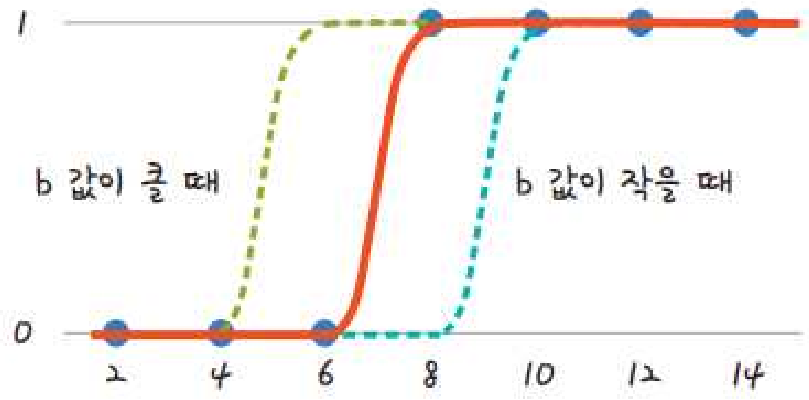
▼ **그림 6-3 | a 값이 클 때와 작을 때의 그래프 변화**



2 시그모이드 함수

- **시그모이드 함수**
  - b는 그래프의 좌우 이동을 의미
  - 그림 6-4와 같이 b 값이 크고 작아짐에 따라 그래프가 이동

▼ **그림 6-4 | b 값이 클 때와 작을 때의 그래프 변화**

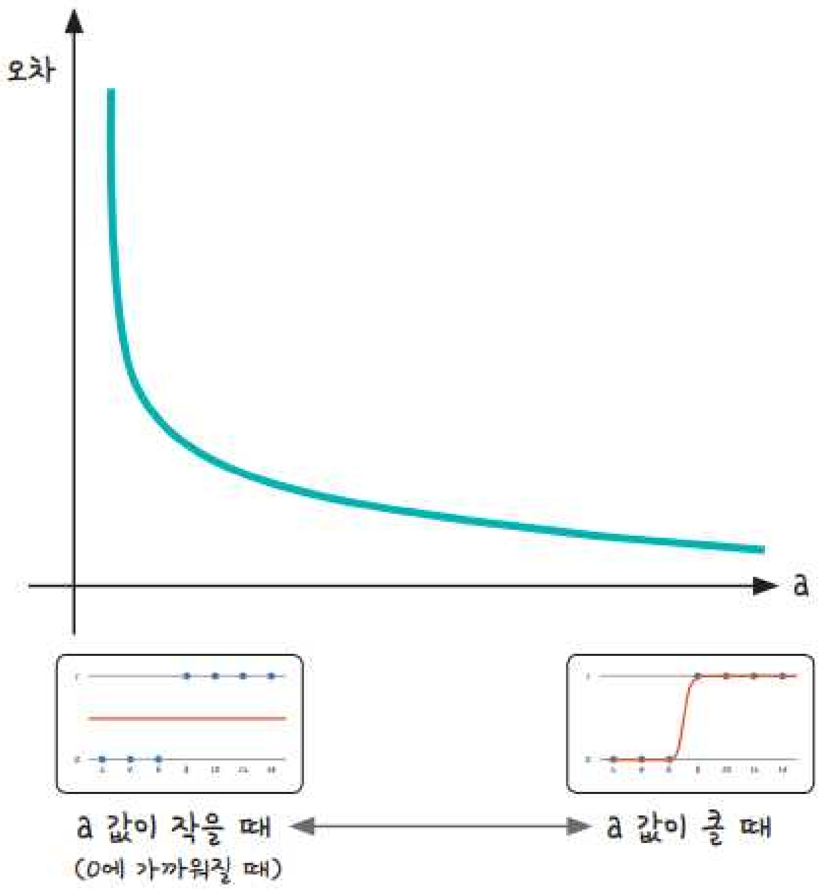


2 시그모이드 함수

- **시그모이드 함수**
  - a 값과 b 값에 따라 오차가 변함
  - a 값에 따라 변화하는 오차를 그래프로 나타내면 그림 6-5와 같음

2 시그모이드 함수

▼ 그림 6-5 | a와 오차의 관계: a가 작아질수록 오차는 무한대로 커지지만, a가 커진다고 해서
오차가 없어지지는 않는다

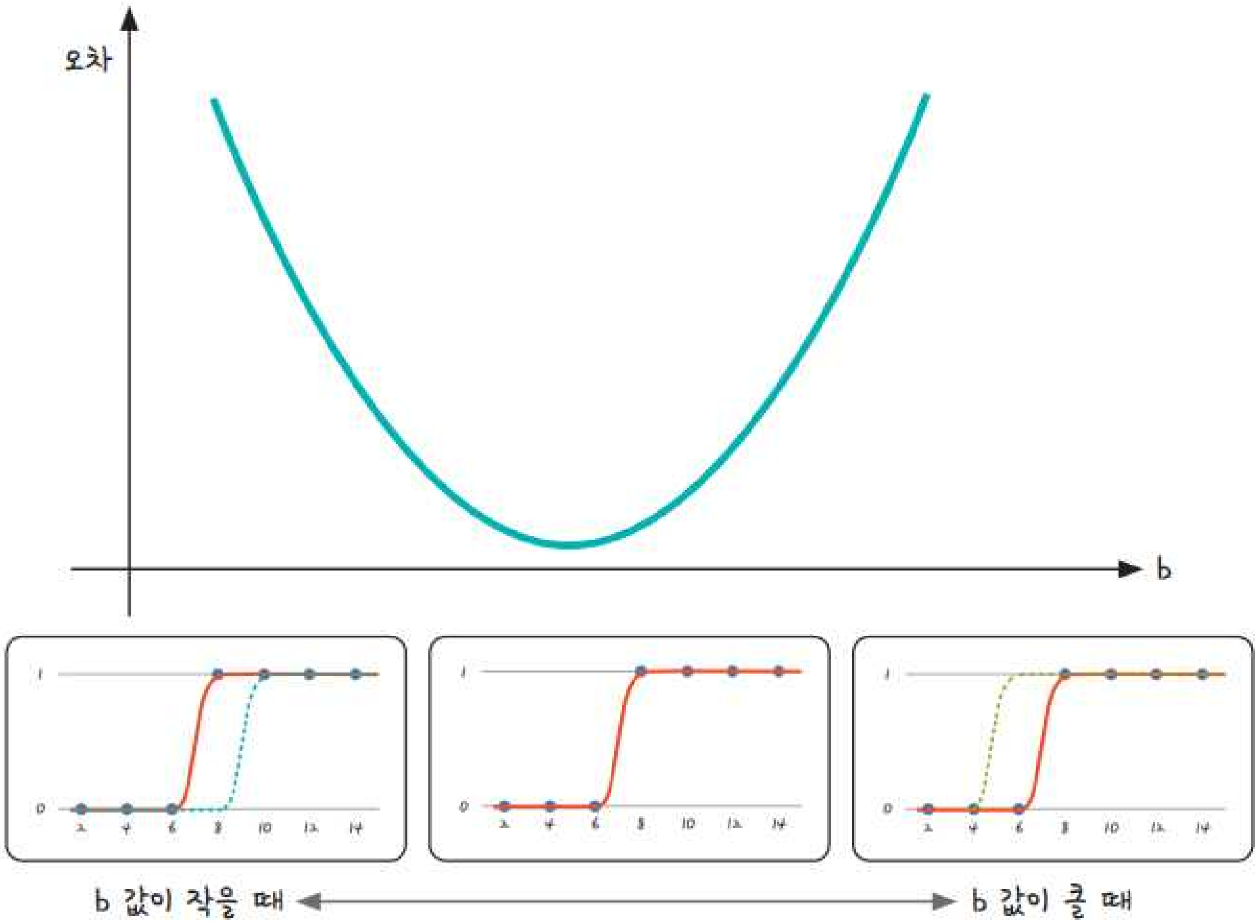


2 시그모이드 함수

- **시그모이드 함수**
  - a 값이 작아지면 오차는 무한대로 커짐
  - a 값이 커진다고 해서 오차가 무한대로 커지지는 않음

2 시그모이드 함수

▼ 그림 6-6 | b와 오차의 관계: b 값이 너무 작아지거나 커지면 오차도 이에 따라 커진다



2 시그모이드 함수

- **시그모이드 함수**
  - b 값이 너무 크거나 작을 경우 오차는 그림 6-6과 같이 이차 함수 그래프와 유사한 형태로 나타남

3 오차 공식

3 오차 공식

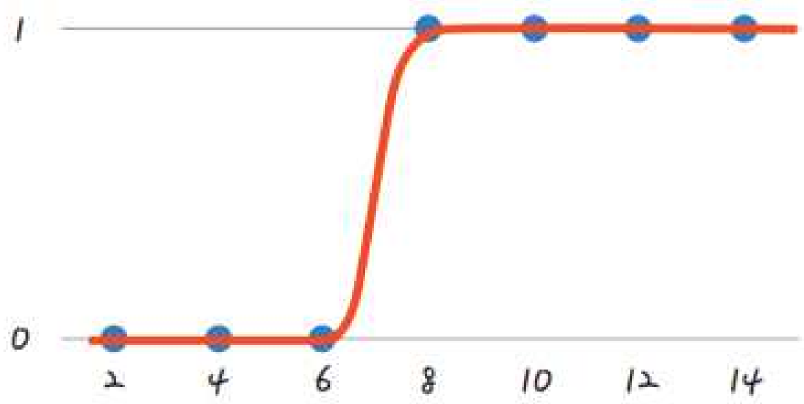
- **오차 공식**
  - 이제 우리에게 주어진 과제는 또다시 a 값과 b 값을 구하는 것임을 알았음
  - 시그모이드 함수에서 a 값과 b 값을 어떻게 구해야 할까?
  - 답은 역시 경사 하강법
  - 경사 하강법은 먼저 오차를 구한 후 오차가 작은 쪽으로 이동시키는 방법이라고 했음
  - 이번에도 예측 값과 실제 값의 차이, 즉 오차를 구하는 공식이 필요함
  - 이번에도 앞서 배웠던 평균 제곱 오차를 사용하면 될까?
  - 안타깝게도 이번에는 평균 제곱 오차를 사용할 수 없음

3 오차 공식

오차 공식

- 오차 공식을 도출하기 위해 시그모이드 함수 그래프의 특징을 다시 한 번 살펴보자

▼ **그림 6-7 | 시그모이드 함수 그래프**



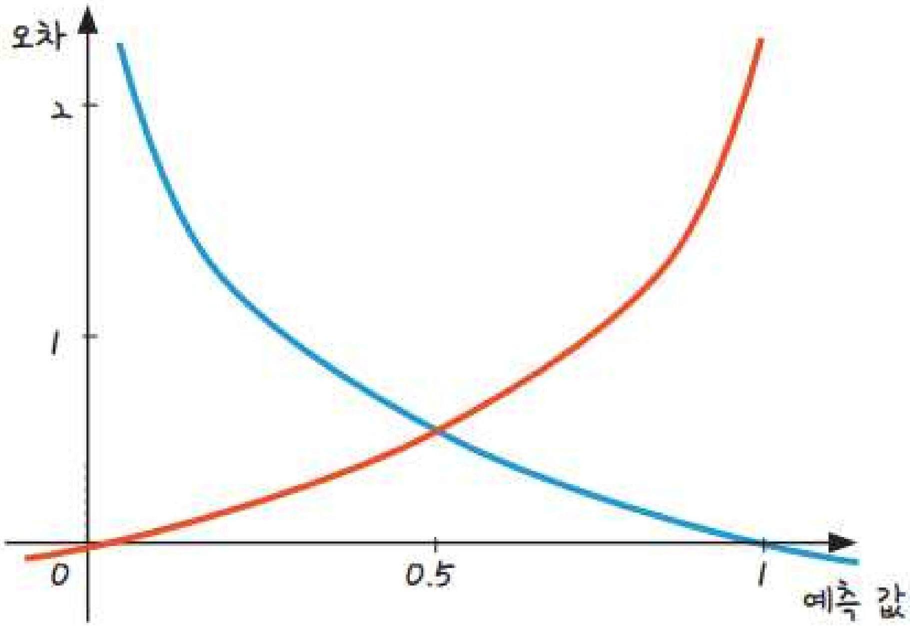
- 시그모이드 함수의 특징은 y 값이 0과 1 사이라는 것
- 실제 값이 1일 때 예측 값이 0에 가까워지면 오차가 커짐
- 반대로 실제 값이 0일 때 예측 값이 1에 가까워지는 경우에도 오차는 커짐
- 이를 공식으로 만들 수 있게 하는 함수가 바로 로그 함수

4 로그 함수

4 로그 함수

- **로그 함수**
  - '3장. 딥러닝을 위한 기초 수학'의 로그 함수에서 배운 그래프를 다시 가져오겠음

▼ **그림 6-8 | 실제 값이 1일 때(파란색)와 0일 때(빨간색) 로그 함수 그래프**



4 로그 함수

- **로그 함수**
  - 파란색 선은 실제 값이 1일 때 사용할 수 있는 그래프
  - 예측 값이 1일 때 오차가 0이고, 반대로 예측 값이 0에 가까울수록 오차는 커짐
  - 빨간색 선은 반대로 실제 값이 0일 때 사용할 수 있는 함수
  - 예측 값이 0일 때 오차가 없고, 1에 가까워질수록 오차가 매우 커짐

4 로고 함수

로그 함수

- 앞의 파란색과 빨간색 그래프의 식은 각각 -logh와 -log(1-h)
- 실제 값이 1일 때는 -log h 그래프를 쓰고, 0일 때는 -log(1-h) 그래프를 써야 함
- 이는 다음과 같은 방법으로 해결할 수 있음

$$-\{\underbrace{y\log h}_{A}+\underbrace{(1-y)\log(1-h)}_{B}\}$$

4 로그 함수

- **로그 함수**
  - 실제 값을 y라고 할 때, 이 값이 1이면 B 부분이 없어짐
  - 반대로 0이면 A 부분이 없어짐
  - 실제 값에 따라 빨간색 그래프와 파란색 그래프를 각각 사용할 수 있음
  - 이렇게 해서 평균 제곱 오차를 대체할 만한 손실 함수를 구함
  - 이 함수를 머신 러닝에서는 **교차 엔트로피 오차**(cross entropy error) 함수라고 함
  - 즉, 선형 회귀에서는 평균 제곱 오차 함수를, 로지스틱 회귀에서는 교차 엔트로피 오차 함수를 사용하게 되는 것
  - 이 두 가지 함수에서 출발해 지금은 더 다양한 손실 함수들이 존재
  - 이와 관련해서는 '10장. 딥러닝 모델 설계하기'에서 다시 다룰 것

5 텐서플로에서 실행하는 로지스틱 회귀 모델

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - 텐서플로에서 실행하는 방법은 앞서 선형 회귀 모델을 만들 때와 유사함
  - 다른 점은 오차를 계산하기 위한 손실 함수가 평균 제곱 오차 함수에서 크로스 엔트로피 오차로 바뀐다는 것
  - 먼저 표 6-1에서 본 합격 여부 데이터를 다음과 같이 만들어 주자

```
x = np.array([2, 4, 6, 8, 10, 12, 14])
y = np.array([0, 0, 0, 1, 1, 1, 1])
```

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - 이제 모델을 준비
  - 먼저 시그모이드 함수를 사용하게 되므로 activation 부분을 sigmoid로 바꾸어 줌

```
model.add(Dense(1, input_dim=1, activation='sigmoid'))
```

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - 손실 함수로 교차 엔트로피 오차 함수를 이용하기 위해 loss를 binary_crossentropy로 설정

```
model.compile(optimizer='sgd', loss='binary_crossentropy')
```

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - model.predict(x) 함수를 이용해 학습 시간 x가 입력되었을 때의 결과를 그래프로 그려 보면 다음과 같음

```
plt.scatter(x, y)
plt.plot(x, model.predict(x), 'r')
plt.show()
```

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - 임의의 학습 시간에 따른 합격 확률을 보여 주는 부분을 다음과 같이 설정

```
hour = 7
prediction = model.predict([hour])
print("%.f시간을 공부할 경우, 합격 예상 확률은 %.01f%%입니다." % (hour,
prediction * 100))
```

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - 모두 정리하면 다음과 같음

실습 | 텐서플로에서 실행하는 다중 선형 회귀

```
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

x = np.array([2, 4, 6, 8, 10, 12, 14])
y = np.array([0, 0, 0, 1, 1, 1, 1])

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

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**

```
# 교차 엔트로피 오차 함수를 이용하기 위해 'binary_crossentropy'로 설정합니다.
model.compile(optimizer='sgd', loss='binary_crossentropy')
model.fit(x, y, epochs=5000)

# 그래프로 확인해 봅니다.
plt.scatter(x, y)
plt.plot(x, model.predict(x), 'r')
plt.show()

# 임의의 학습 시간을 집어넣어 합격 예상 확률을 예측해 보겠습니다.
hour = 7
prediction = model.predict([hour])
print("%.f시간을 공부할 경우, 합격 예상 확률은 %.01f%%입니다." % (hour,
prediction * 100))
```

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**

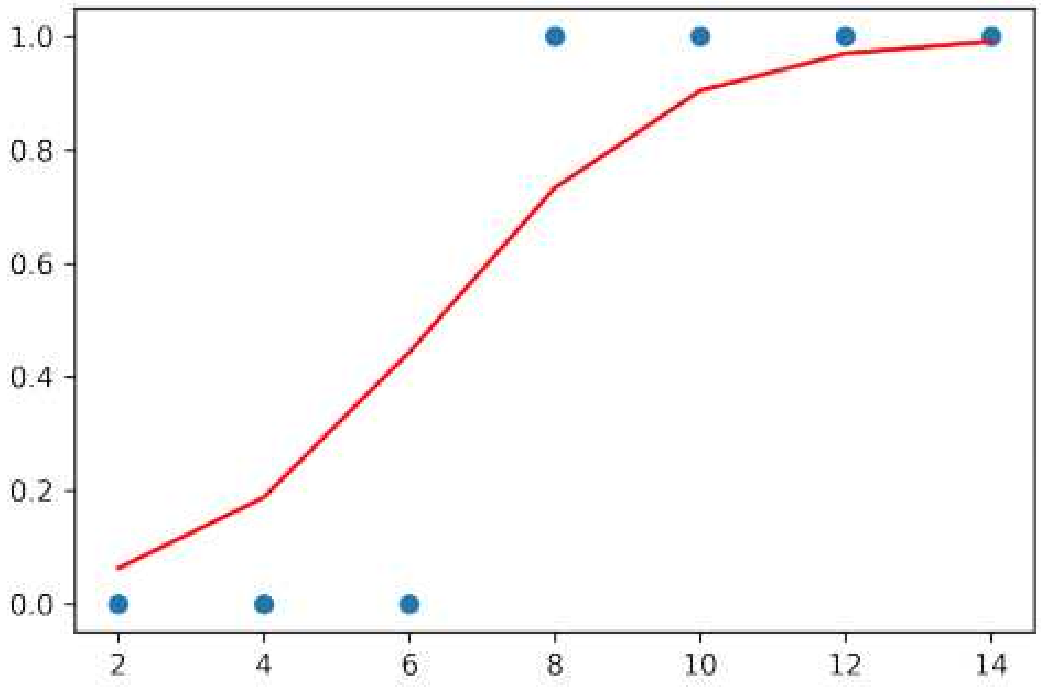
실행 결과

```
Epoch 1/5000
1/1 [==============================] - 0s 183ms/step - loss: 0.5741
... (중략) ...
Epoch 5000/5000
1/1 [==============================] - 0s 1ms/step - loss: 0.1882
```

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- 텐서플로에서 실행하는 로지스틱 회귀 모델

▼ 그림 6-9 | 실행 결과



7시간을 공부할 경우, 합격 예상 확률은 59.7%입니다.

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - 출력되는 그래프는 학습이 진행됨에 따라 점점 시그모이드 함수 그래프의 형태를 취해 가는 것을 보여 줌
  - 학습된 모델의 테스트를 위해 여러 가지 임의의 시간을 집어넣고 테스트해 보면, 학습 시간이 7보다 클 경우에는 합격 확률이 50%가 넘는다는 것을 알 수 있음
  - 데이터양이나 학습 시간 등 환경에 따라 예측 정확도는 더욱 향상될 수 있음

5 텐서플로에서 실행하는 로지스틱 회귀 모델

- **텐서플로에서 실행하는 로지스틱 회귀 모델**
  - 지금까지 선형 회귀와 로지스틱 회귀를 사용한 모델링에 관해 알아보았음
  - 이 두 가지가 딥러닝의 기본 요소가 된다는 것은 이미 설명한 바 있음
  - 이러한 통계 모델링은 어떻게 해서 딥러닝과 연관을 갖게 될까?
  - 로지스틱 회귀 모델의 전신인 퍼셉트론과 퍼셉트론의 한계를 극복하며 탄생한 신경망에 대해 상세히 알아보며 딥러닝 학습 속도를 더해 보자