선형 회귀

선형회귀

- 지도학습에는 회귀(regression)와 분류(classification)가 있다.
- 회귀 : 연속적인 값을 예측
- 분류 : 입력을 어떤 카테고리 중의 하나로 예측

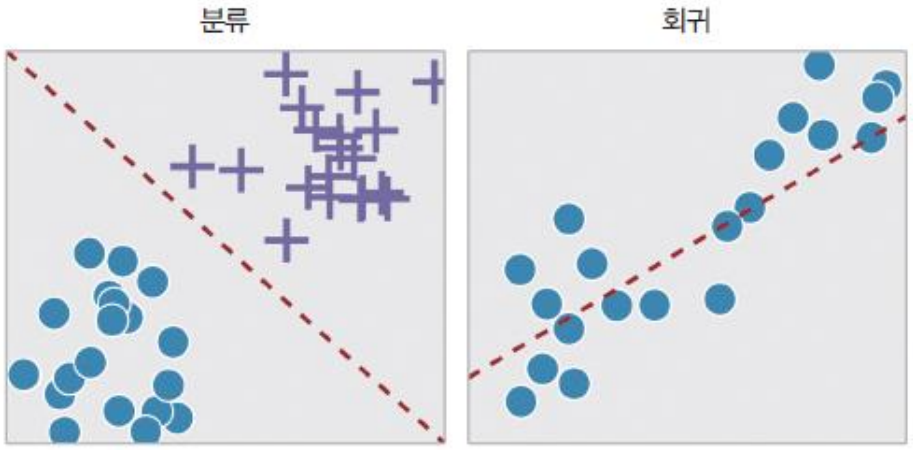


그림 4-1 회귀와 분류

선형 회귀

- 회귀(regression)
  - 데이터들을 2차원 공간에 찍은 후에 이들 데이터들을 가장 잘 설명하는 직선이나 곡선을 찾는 문제
  - y = f(x)에서 출력 y가 실수이고 입력 x도 실수일 때 함수 f(x)를 예측하는 것
  - 주식 가격 예측, 온도 변화, 전력 수요 변동 등의 연속적인 값을 예측

그림 4-2 회귀의 종류

선형 회귀

- 선형 회귀(linear regression) : 선형 모델을 사용하는 회귀 문제
  - 부모의 키와 자녀의 키의 관계 조사
  - 면적에 따른 주택의 가격
  - 연령에 따른 실업율 예측
  - 공부 시간과 학점 과의 관계
  - CPU 속도와 프로그램 실행 시간 예측

- 사이트 소개 : desmos

선형 회귀 소개

- 직선의 방정식: f(x) = mx + b
- 입력 데이터를 가장 잘 설명하는 기울기와 절편값을 찾는 문제
- 머신러닝에서는 기울기 대신에 가중치(weight), 절편 대신에 바이어스(bias)라고 한다.
- 선형 회귀의 기본식: f(x) = wx + b

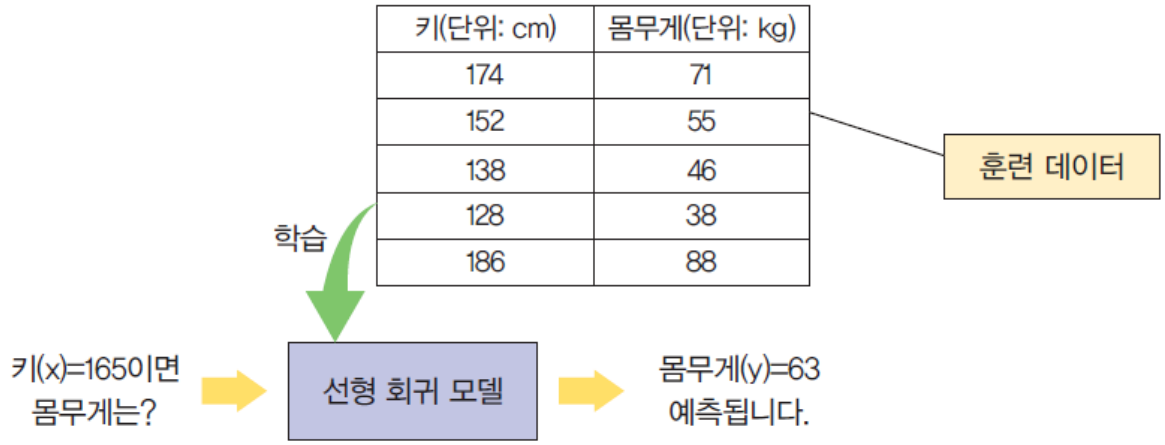


그림 4-3 선형 회귀의 예제

선형 회귀의 종류

- 단순 선형 회귀 : 독립 변수(x)가 하나

$$f(x) = wx + b$$

- 다중 선형 회귀 : 독립 변수가 여러 개

$$f(x, y, z) = w_0 + w_1 x + w_2 y + w_3 z$$

$$\text{매출} = w_0 + w_1 \times \text{인터넷 광고} + w_2 \times \text{신문광고} + w_3 \times TV\text{광고}$$

선형 회귀의 원리

- 학습 데이터와 가장 잘 맞는 하나의 직선을 찾는 것

x	y
1	2
2	5
3	6

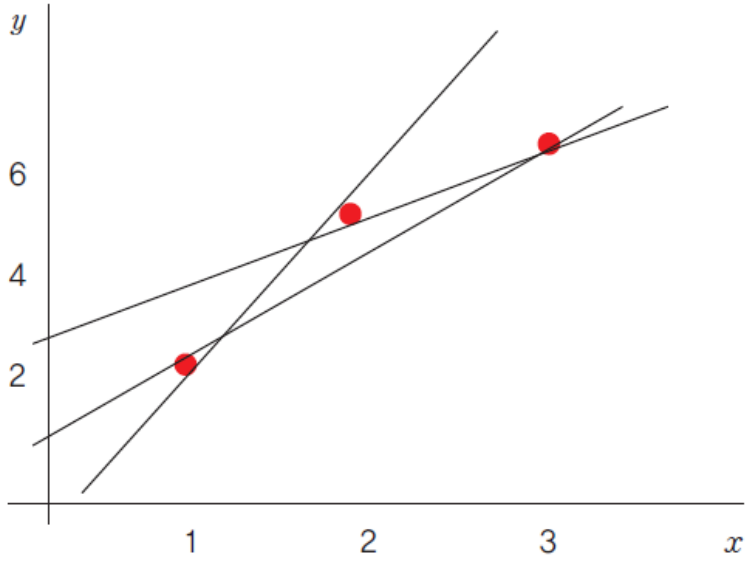


그림 4-4 데이터와 직선

직선과 데이터의 거리

- 간격이 적을수록 직선이 데이터와 잘 맞는다고 할 수 있다. 간격들은 음수일 제곱하여 수 있으니 간격의 제곱을 하는 것이 좋다.
- 직선과 데이터 사이의 간격을 합한 값을 손실 함수(loss function) 또는 비용 함수(cost function)라고 한다.

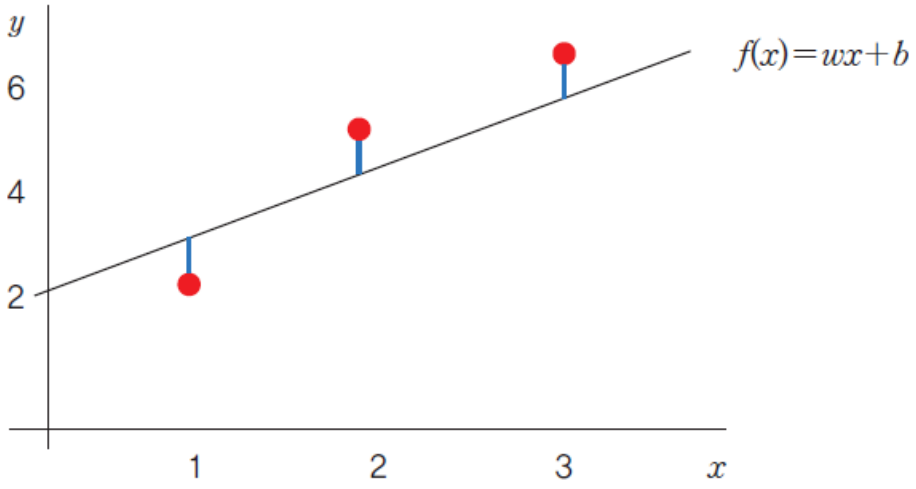


그림 4-5 데이터와 직선 간의 거리

손실 함수

- 평균제곱오차(MSE. Mean Squared Error)

$$Loss = \frac{1}{3}\left((f(x_1) - y_1)^2 + (f(x_2) - y_2)^2 + (f(x_3) - y_3)^2 \right)$$

일반화

$$Loss(W, b) = \frac{1}{n}\sum_{i=1}^{n} (f(x_i) - y_i)^2 = \frac{1}{n}\sum_{i=1}^{n} ((Wx_i + b) - y_i)^2$$

우리가 찾는 것

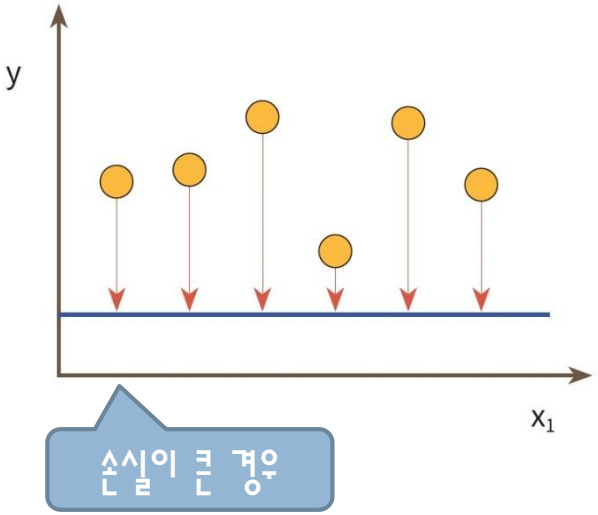
$$\underset{W,b}{\operatorname{argmin}}\, Loss(W, b)$$

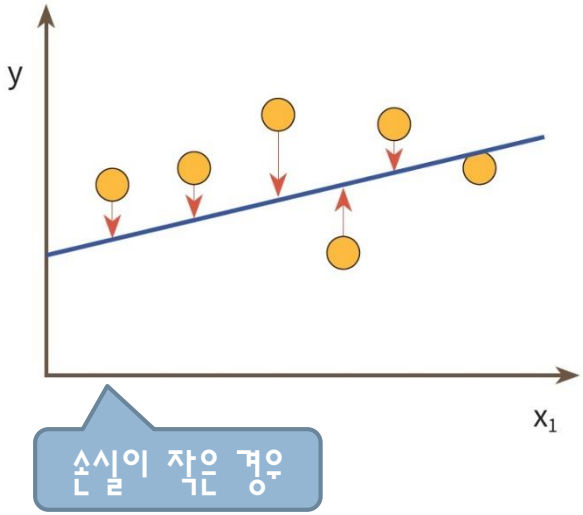
n = 훈련 데이터의 개수

- 손실함수 값을 최소로 하는 w와 b를 찾는 것 -> 이것을 학습이라 한다.

학습과 손실

- 머신러닝에서 모델을 학습시킨다는 것은 훈련 데이터로부터 손실을 최소화하는 가중치(w)와 바이어스(b)을 학습(결정)하는 것
- 손실(loss) : 잘못된 예측에 대한 벌점. 특정 샘플에서 모델의 예측이 얼마나 잘못되었는지를 나타내는 크기
- 모델 학습의 목표 : 모든 샘플에서 평균적으로 가장 작은 손실을 갖는 가중치와 바이어스를 찾는 것





선형 회귀에서 손실 함수 최소화 방법

- 머신러닝 알고리즘은 손실 함수의 값이 최소화되는 방향을 찾아서 w와 b를 변경하는 작업을 반복하게 된다. 손실 함수가 0이 되거나 0에 가까운 값이 되면 학습이 종료된다.

- 손실 함수의 값이 작아지는 방향을 어떻게 알 수 있을까?

경사 하강법(gradient descent method)

- 가중치를 손실이 줄어드는 방향으로 움직이도록 현재 위치에서 손실 함수의 기울기(경사)를 계산하는 방법

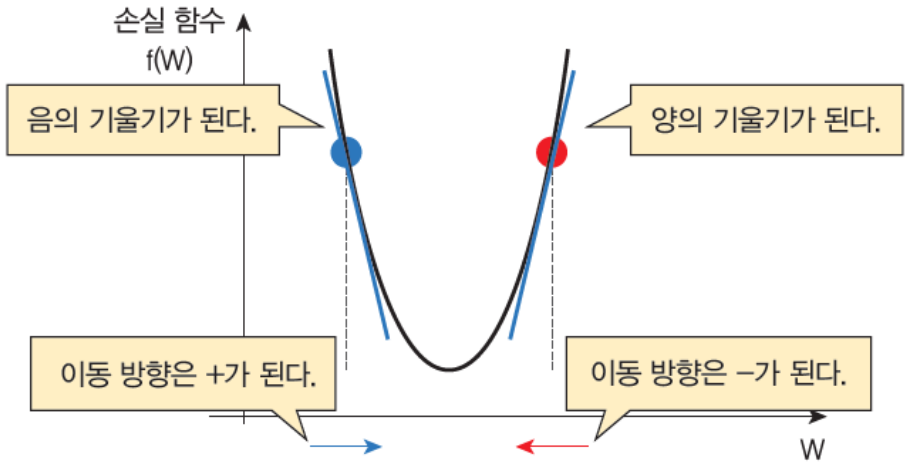
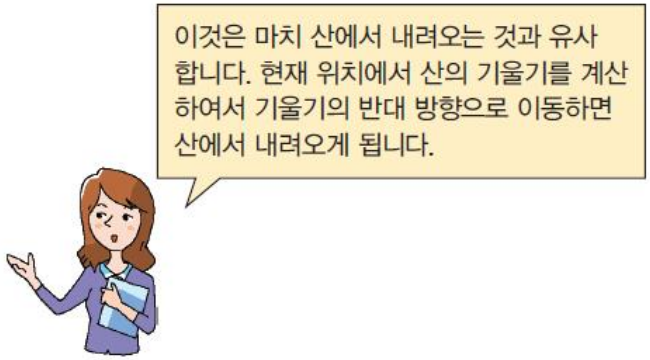


그림 4-7 경사 하강법의 이해

- 기울기가 양수로 계산되었다면, 가중치를 증가했을 때 손실이 높아진다.
- 기울기가 음수로 계산되었다면, 가중치를 증가했을 때 손실이 낮아진다.
- 따라서 손실 함수를 줄이기 위해서는 현재의 위치에서 손실 함수의 기울기를 계산하여서 기울기의 반대 방향으로 이동하면 된다.

경사 하강법


기울기가 음수이니
오른쪽으로 많이 이동
기울기가 음수이지만 작으므로
오른쪽으로 천천히 이동
목표 도달!


이것은 마치 산에서 내려오는 것과 유사합니다. 현재 위치에서 산의 기울기를 계산하여서 기울기의 반대 방향으로 이동하면 산에서 내려오게 됩니다.

경사 하강법

- 반복적인 접근방식을 사용한다.
- 기울기가 음수이면 매개변수(w)는 증가시키고, 반대로 기울기가 양수이면 w는 감소시킨다.
- w가 업데이트되면 새로운 손실값을 얻을 수 있다. 손실 함수를 다시 계산하고 기울기도 다시 계산한다. 이렇게 기울기의 반대 방향으로 가다 보면 최저값을 찾을 수 있다.

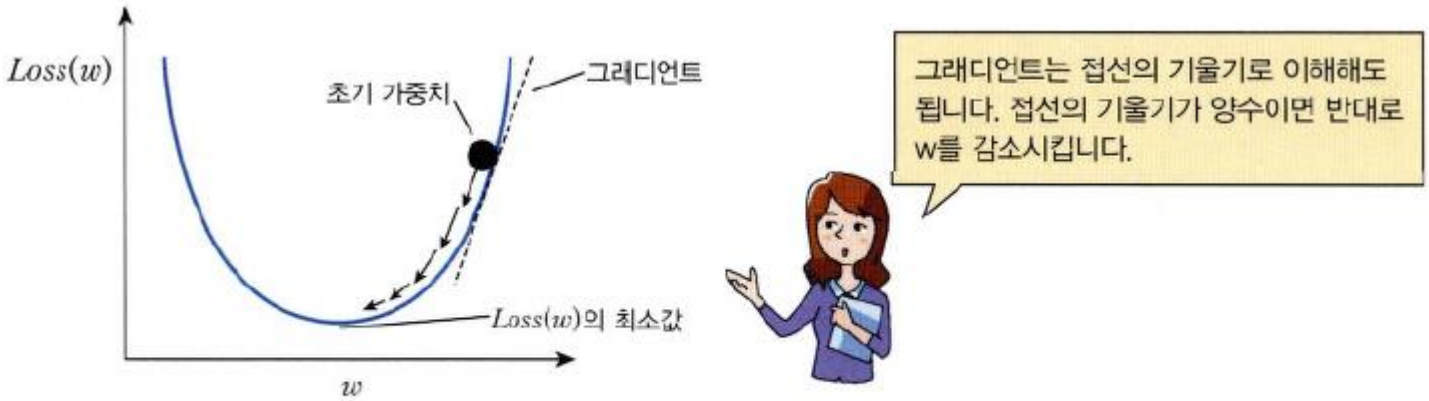


그림 4-8 경사 하강법

학습률

- 기울기의 반대방향으로 w가 한 번에 얼마씩 이동해야 할까?
- 학습률(learning rate) : 한번에 매개변수(w)를 변경하는 비율

- 학습률이 아주 작으면 작은 보폭으로 여러 번 이동. 따라서 계산을 여러 번 해야 최저값에 도달할 수 있다.
- 학습률이 크면 더 큰 보폭으로 이동하게 되고 최소값을 지나쳐서 더 멀리 갈 수도. 최악의 경우 손실이 더 커질 수도. -> 발산

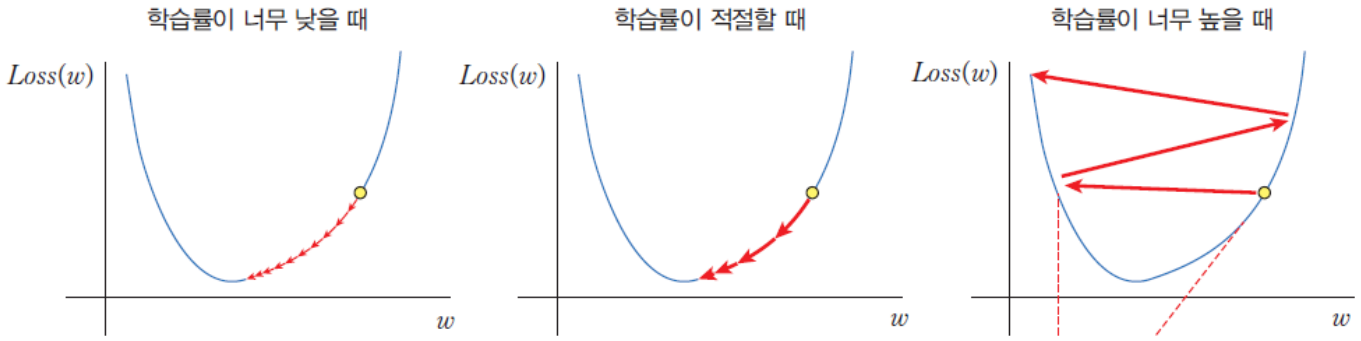


그림 4-9 학습률

지역 최소값 문제

- 출발점(초기값)이 중요함을 확인할 수 있는 경우

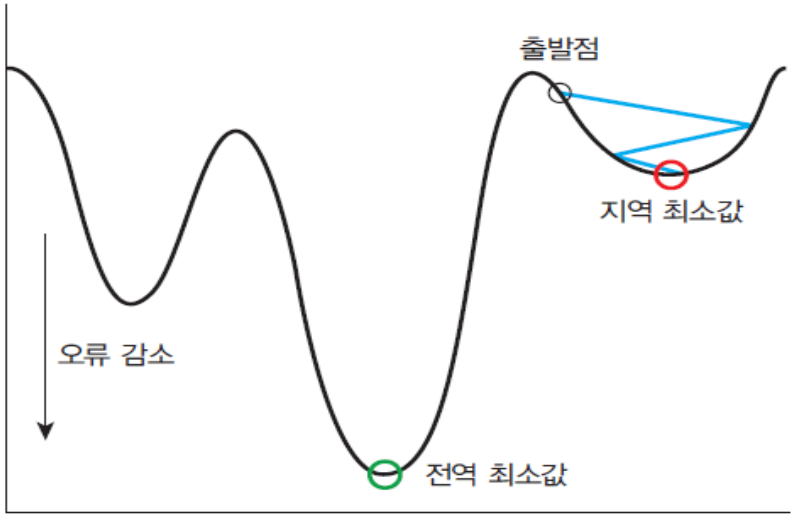


그림 4-10 지역 최소값과 전역 최소값

선형 회귀에서 경사하강법. 편미분

$$Loss(w, b) = \frac{1}{n}\sum_{i=1}^{n}(f(x_i)-y_i)^2 = \frac{1}{n}\sum_{i=1}^{n}((wx_i+b)-y_i)^2$$

(1) 처음에는 w와 b를 모두 0으로 설정한다. 그리고 학습률을 0.01이라고 하자. 학습률은 한 번에 기울기를 변경하는 양이다.

(2) 손실 함수를 w에 대하여 편미분하면 다음과 같은 식을 얻을 수 있다.

$$\frac{\partial Loss(w, b)}{\partial w} = \frac{1}{n}\sum_{i=1}^{n}2((wx_i+b)-y_i)(x_i) = \frac{2}{n}\sum_{i=1}^{n}x_i((wx_i+b)-y_i)$$

(3) 마찬가지로 손실 함수를 b에 대하여 미분하면 다음과 같은 식을 얻을 수 있다.

$$\frac{\partial Loss(w, b)}{\partial b} = \frac{2}{n}\sum_{i=1}^{n}((wx_i+b)-y_i)(1) = \frac{2}{n}\sum_{i=1}^{n}((wx_i+b)-y_i)$$

(4) 우리는 w와 b를 다음과 같이 업데이트한다.

$$w = w-0.01*\frac{\partial Loss}{\partial w}$$

$$b = b-0.01*\frac{\partial Loss}{\partial b}$$

선형 회귀 파이썬 구현 #1

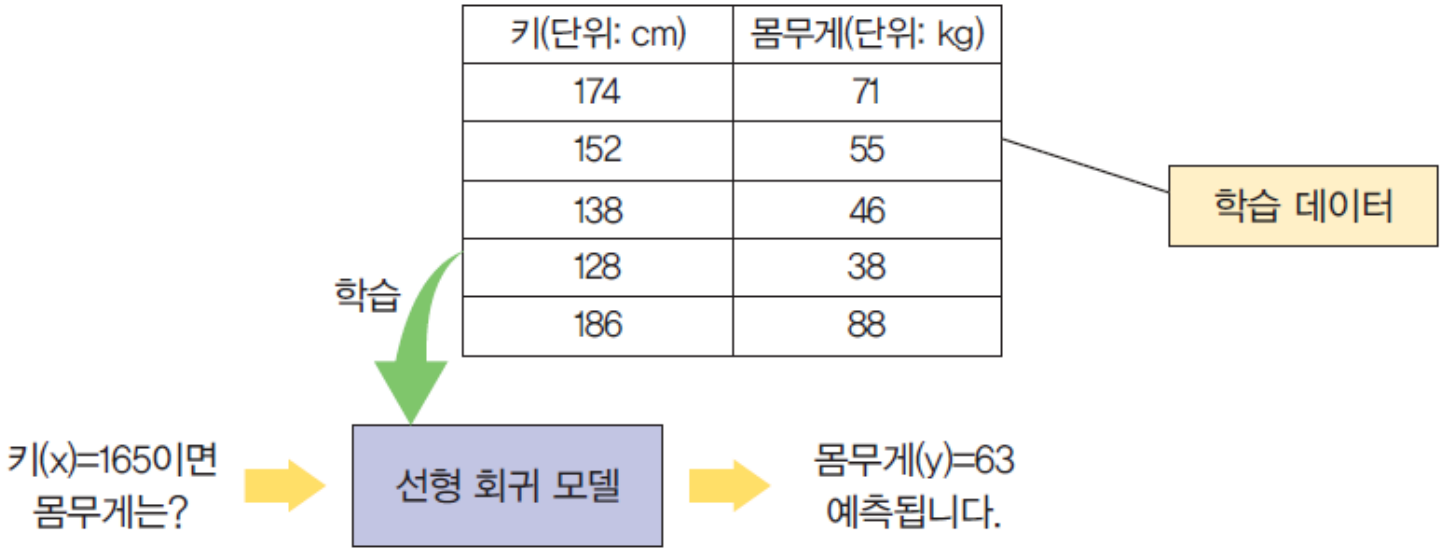
- 넘파이, 파이썬으로 직접 코딩

선형 회귀 파이썬 구현 #2

- 사이킷런(sklearn) 라이브러리를 이용해서 회귀함수 구현
- reg.fit(X, y)

Lab: 선형 회귀 실습

- 인간의 키와 몸무게는 어느 정도 비례할 것으로 예상된다. 아래와 같은 데이터가 있을 때, 선형 회귀를 이용하여 학습시키고 키가 165cm일 때의 예측값을 얻어보자.



과잉 적합 VS 과소 적합

- 과잉 적합(overfitting)
  - 학습하는 데이터에서는 성능이 뛰어나지만 새로운 데이터(일반화)에 대해서는 성능이 잘 나오지 않는 모델을 생성하는 것
  - 한쪽으로 편중된 데이터로 인해 발생
  - 예. 한 종류의 문제집만 계속 반복 풀이. 정작 시험은 성적이 낮아.
  - 다양한 데이터로 학습 필요
- 과소 적합(underfitting)
  - 훈련 데이터에서도 성능이 좋지 않은 경우
  - 모델 자체가 적합지 않은 경우. 더 나은 모델을 찾아야 한다.
  - 데이터가 부족. 특성 자체가 너무 단순해서 학습이 제대로 되지 못함
  - 예. 공은 둥글다 -> 지구, 수박 … 오류

과잉 적합 vs 과소 적합

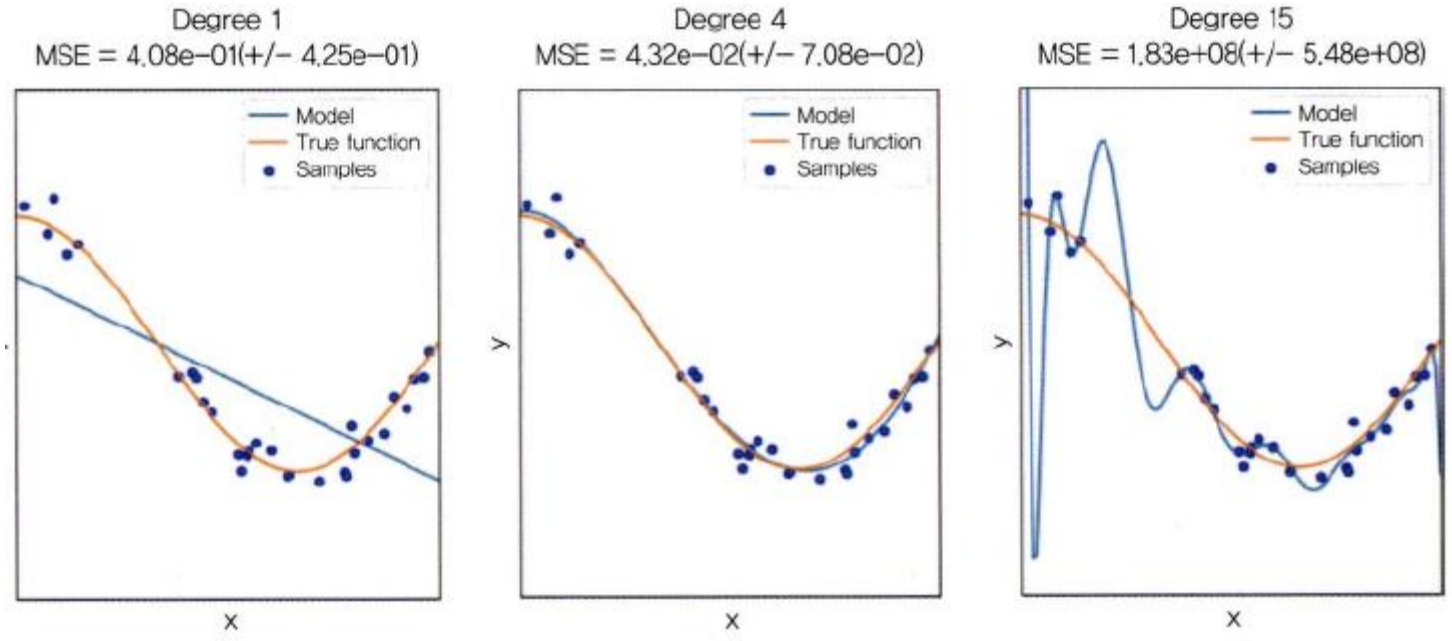
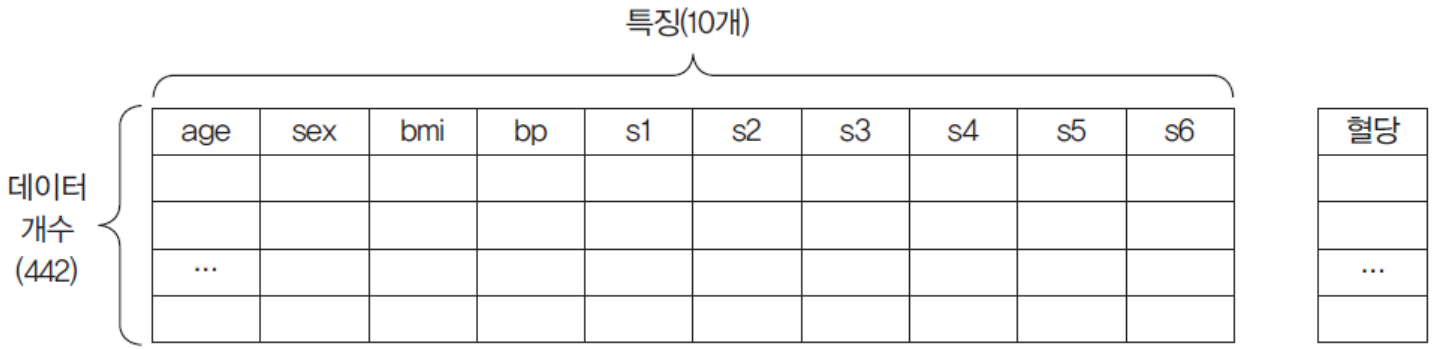


그림 4-11 회귀에서 과잉 적합의 예(출처: 사이킷런 홈페이지)

Lab: 당뇨병 예제

- sklearn 라이브러리에는 당뇨병 환자들의 데이터가 기본적으로 포함되어 있다. BMI하고 혈당치와의 관계를 선형회귀로 분석해보자.



Mini Project: 면적에 따른 집값 예측

- 사용자가 아파트 면적을 입력하면 아파트의 가격이 출력되는 시스템을 만들어보자.