4장 경사하강법

2팀
유지원, 강인영, 김병주, 이윤서

Contents

✔

1회차
Review

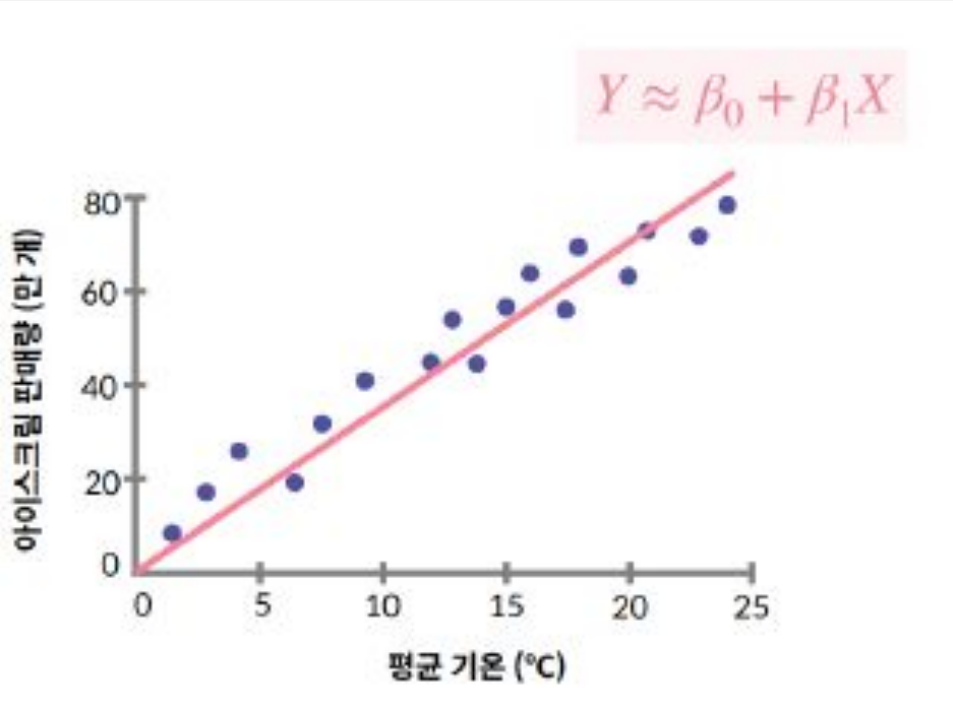
1회차 Review

CNN → RNN → LSTM → GRU

1. 딥러닝 이론 & 선형 회귀란?
2. 최소제곱법
3. 평균 제곱 오차(MSE)

독립변수 여러 개

=> 오차 최소화될 때까지 직선 수정!!



1-1.

Loss(cost) 함수

1-1. Loss(Cost) 함수

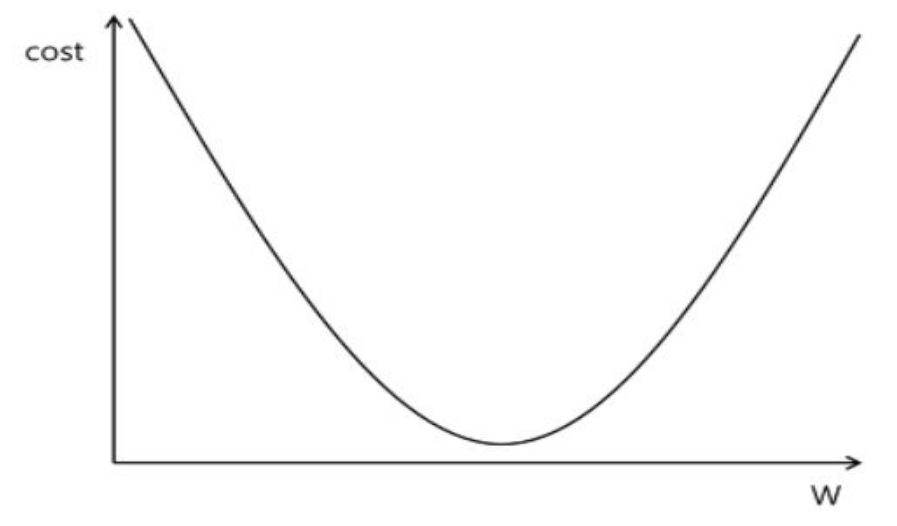
실제 값과 예측 값의 차이의 제곱의 합을 Loss 함수라고 한다! (손실함수 = 비용함수)

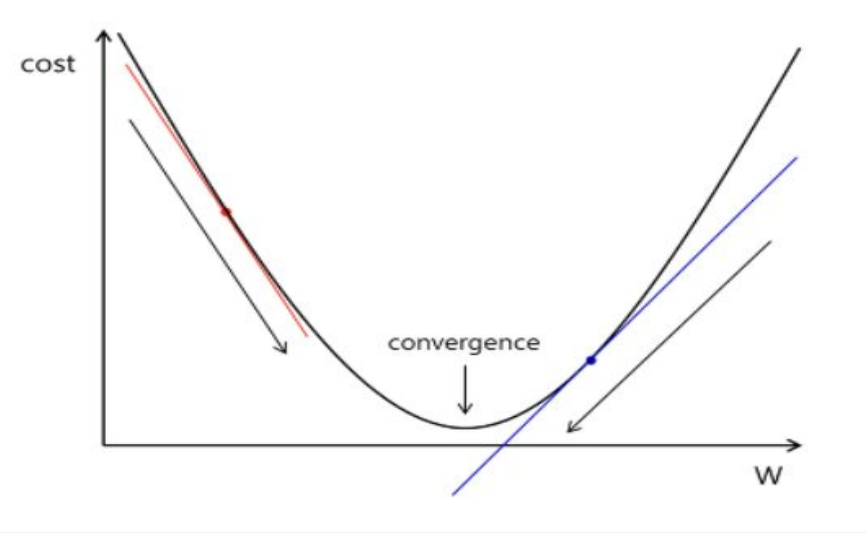
👉 Loss 함수가 작을수록 좋은 모델!

👉 기울기 a, y절편 b 값 조절!

$Y \approx \beta_0 + \beta_1 X$

Loss 함수: $\frac{1}{N}\sum_i^N \left(y^{(i)} - \left(\beta_0 + \beta_1 x^{(i)}\right)\right)^2$ ← ex. MSE





1-1. Loss(Cost) 함수의 종류

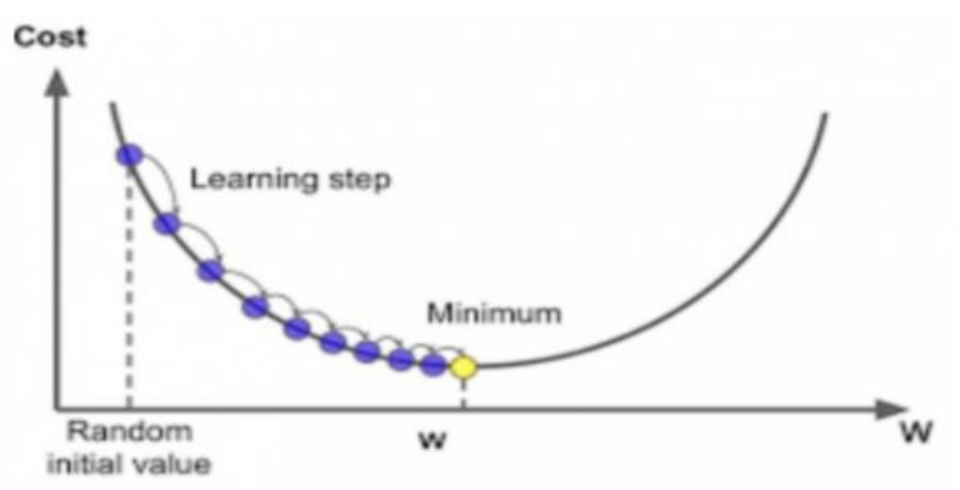
회귀	분류
$-\frac{1}{N}\sum(y-\hat{y})^2$	$-\sum y \log \hat{y}$
MSE => 연속형 변수 예측에 사용됨	Cross-Entropy => y : 실제값(0 or 1) y' : 예측값(확률)

1-1. Loss(Cost) 함수 줄이기

Loss 함수가 최소가 되는 기울기 a, y절편 b 값을 어떻게 찾을까?

$$\underset{\beta_0,\beta_1}{argmin}\frac{1}{N}\sum_{i}^{N}\left(y^{(i)}-\left(\beta_0+\beta_1 x^{(i)}\right)\right)^2$$

1) 경사하강법 <- ❗
2) 정규 방정식(least square)
3) Brute force search
4) etc…



1-2.

단순 선형 회귀에서 경사하강법

1-2. 단순 선형 회귀에서 경사하강법

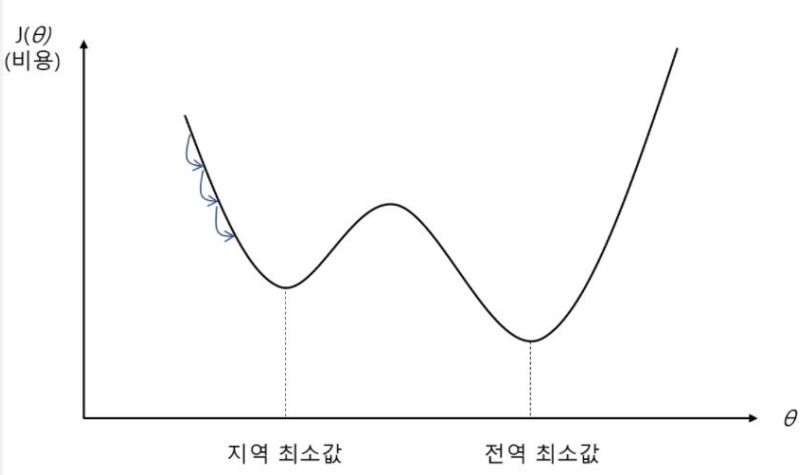
한번의 계산으로 기울기, 절편을 구하는 것이 아니라,
초기값에서 점진적으로 구하는 방식!

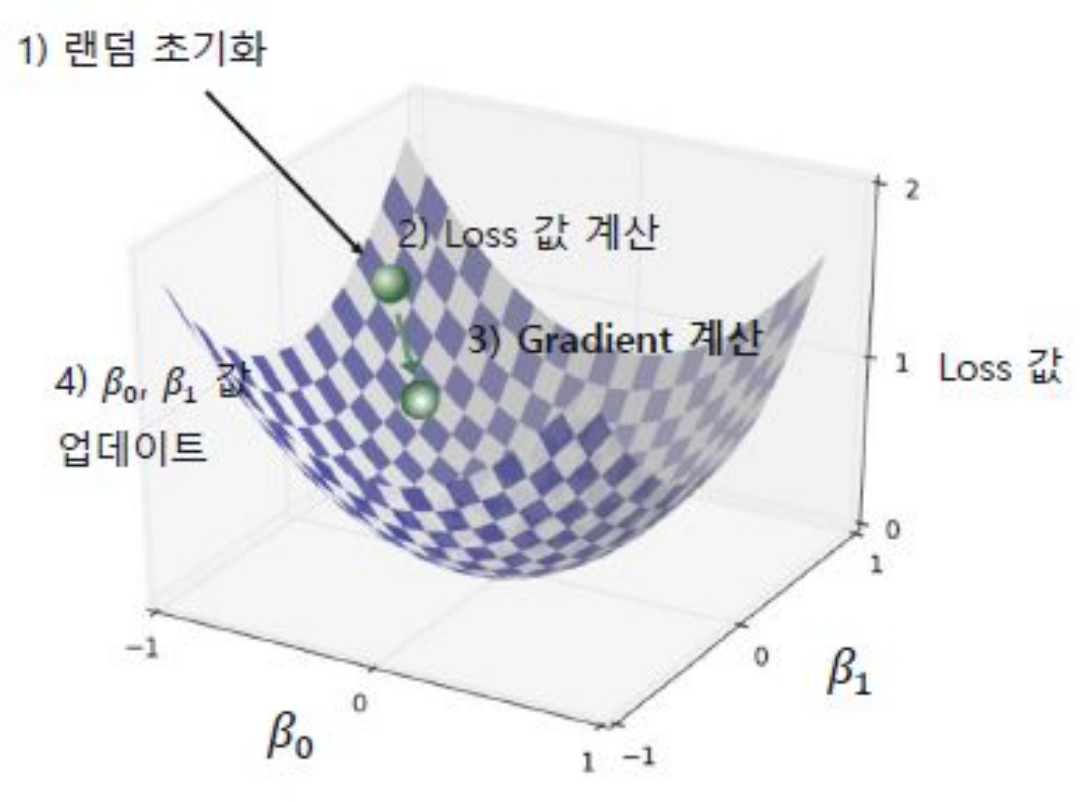
👉 a, b 값을 Loss 함수 값이 작아지게 계속 업데이트~

여기서 잠깐 !!

Loss 함수의 종류는 이차함수 뿐??

->Nope.

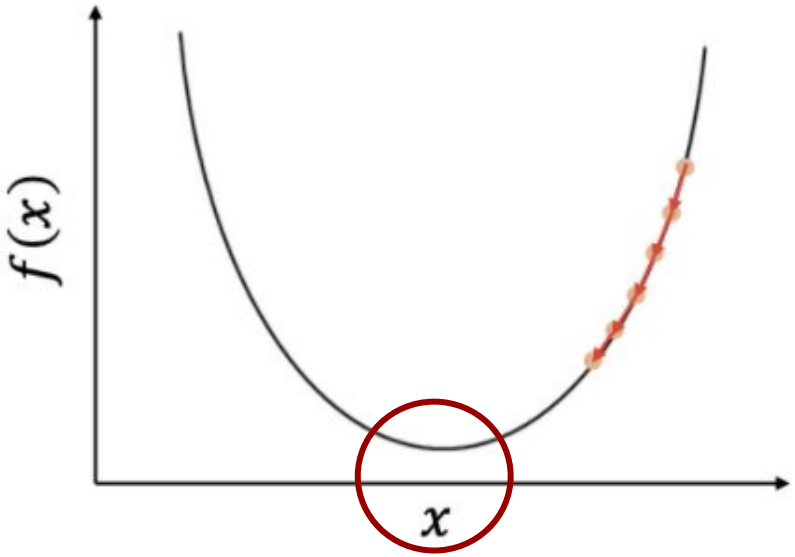




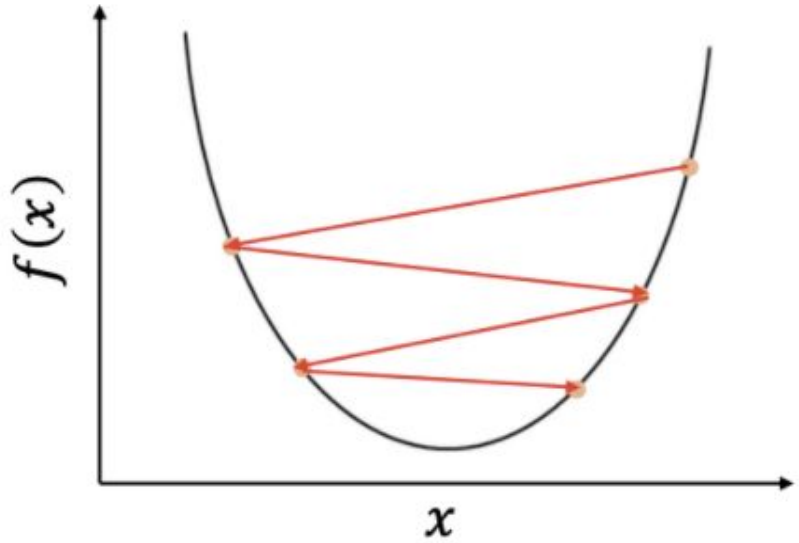
1-2. 단순 선형 회귀에서 경사하강법

학습률 (알파값)

$$x_n = x_{n-1} - \alpha \nabla f(x_{n-1})$$



α가 너무 작은 경우 (수렴이 늦음)

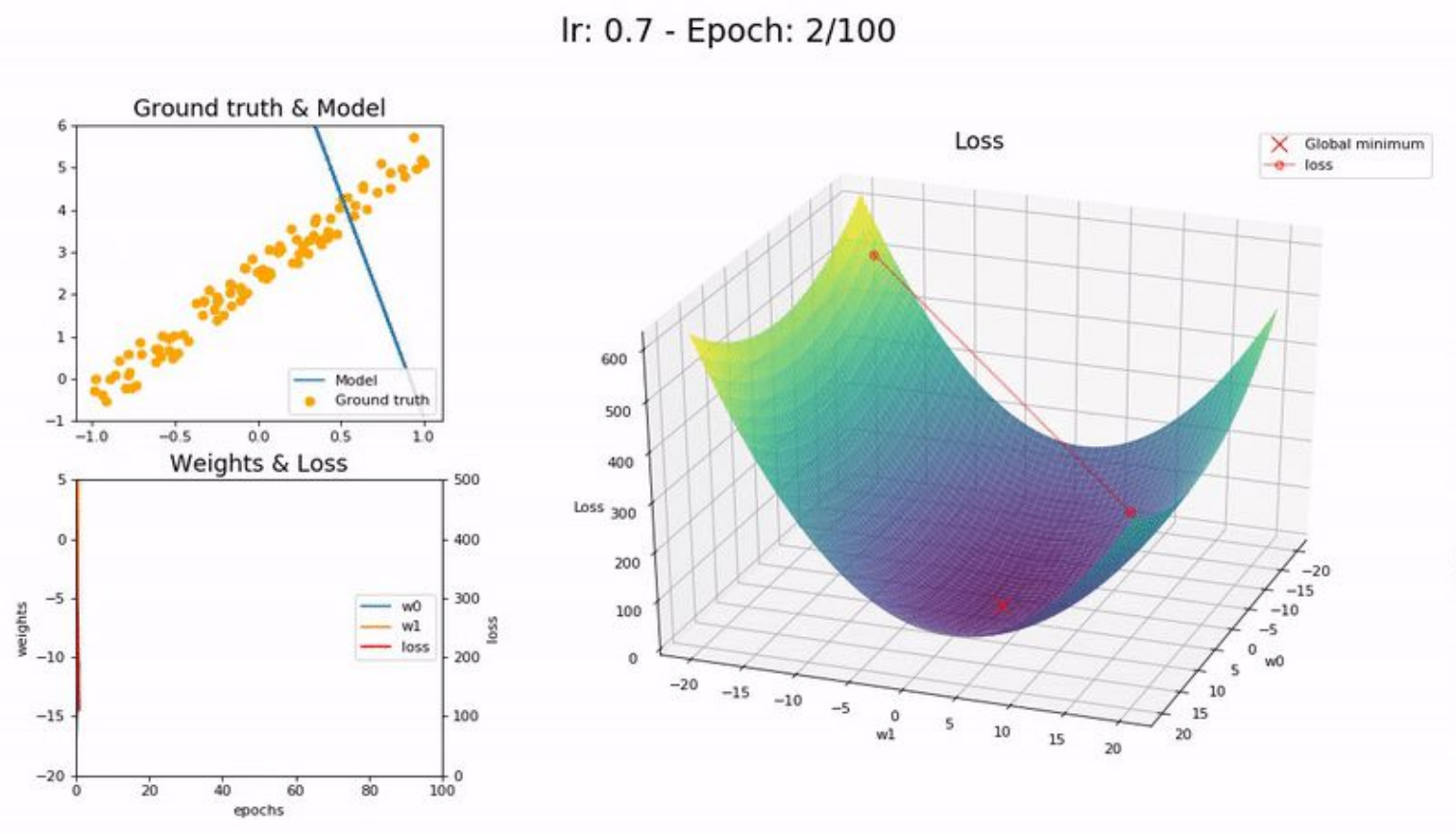


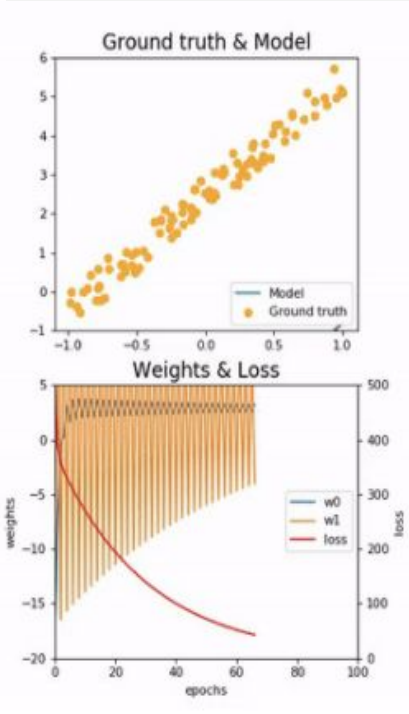
α가 너무 큰 경우 (진동)

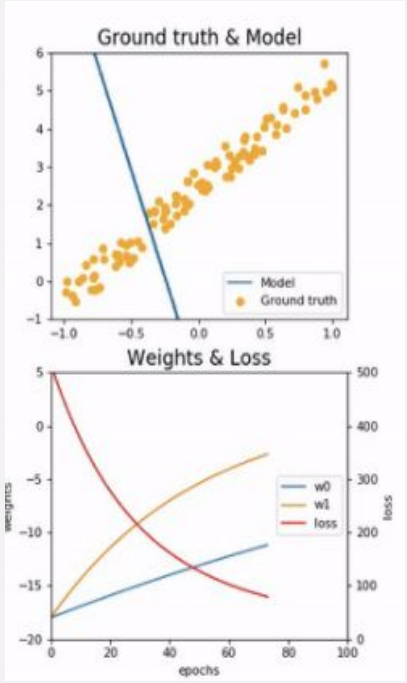
1-2. 단순 선형 회귀에서 경사하강법

학습률 최적화 문제

Lr = 0.99, 0.01 🔽





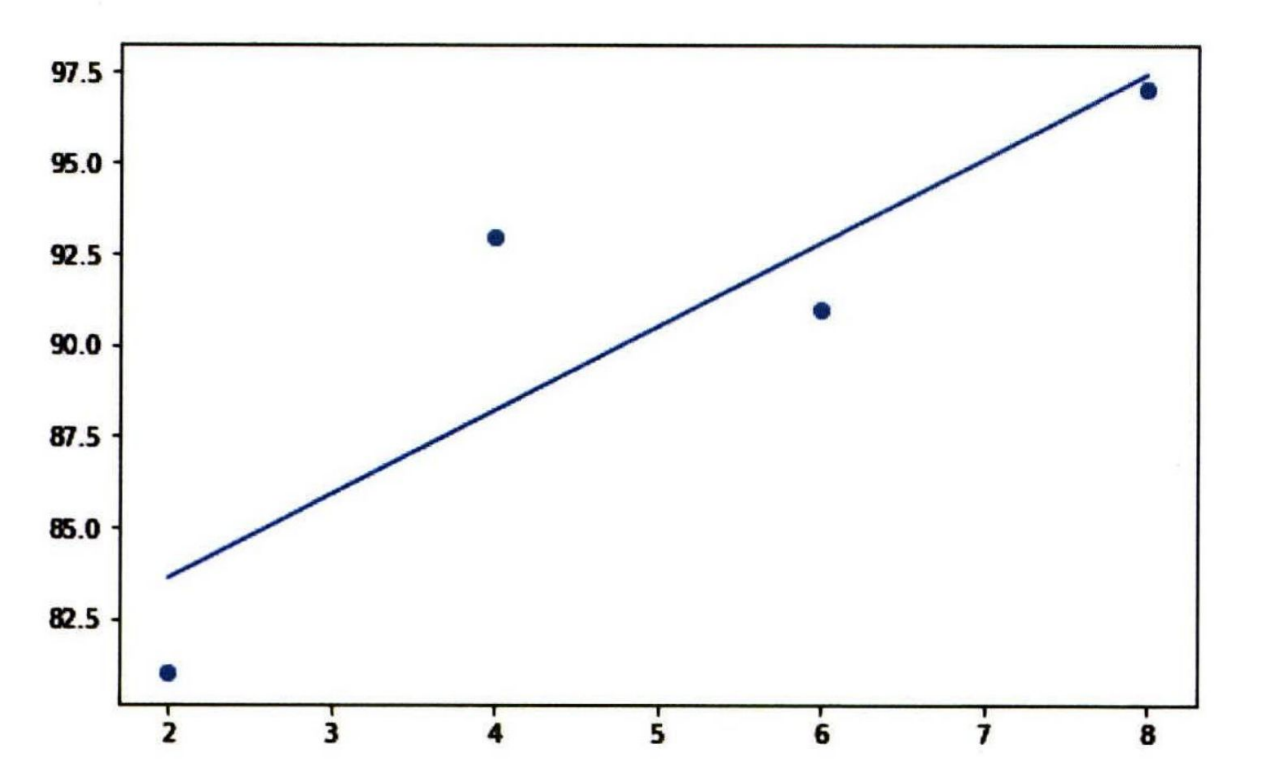


단순 선형 회귀 ver.
경사하강법 실습 타임!

2-1.
다중 선형 회귀에서 경사 하강법

1-2. 다중 선형 회귀에서 경사하강법

단순 선형 회귀의 문제점



1-2. 다중 선형 회귀에서 경사하강법

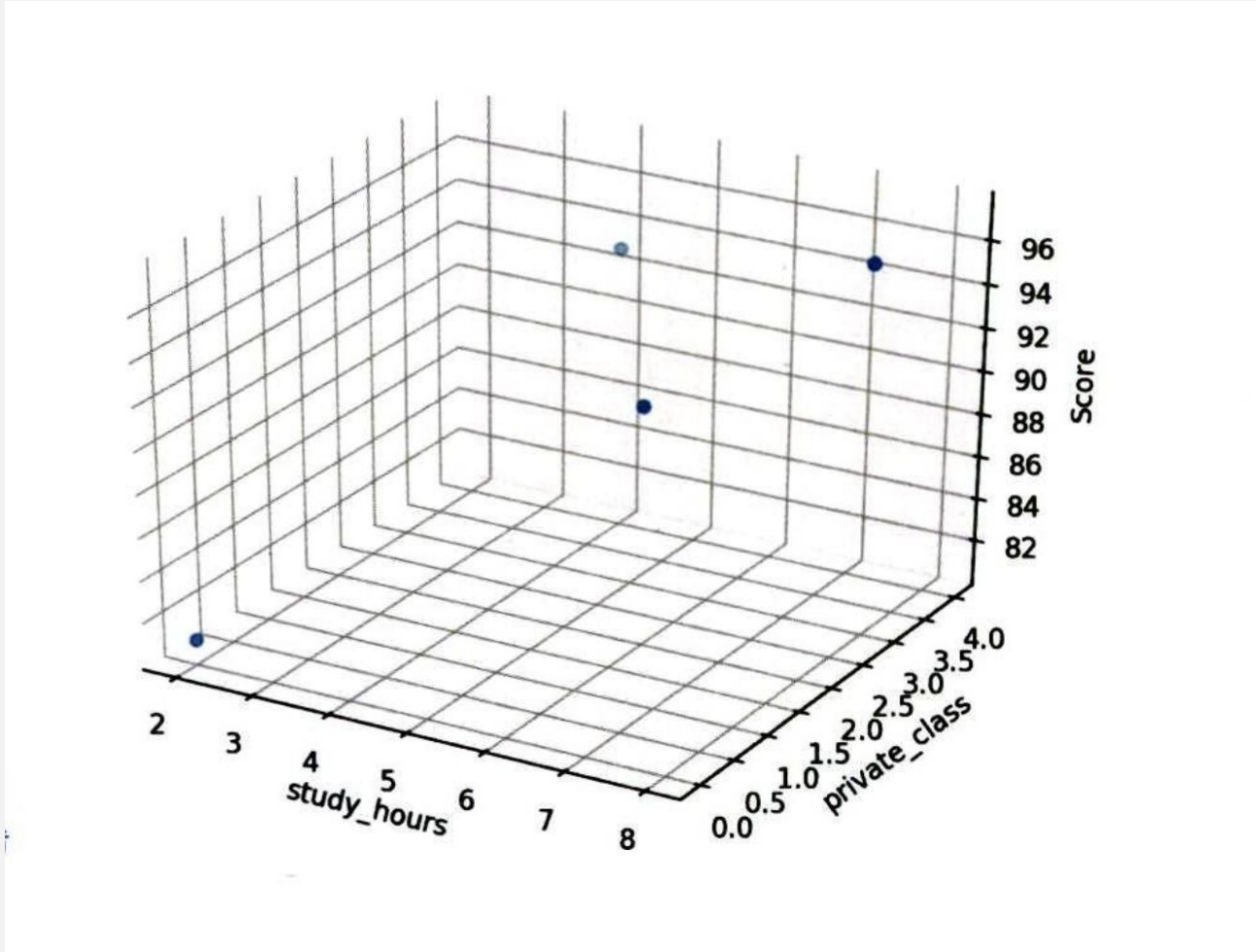
다중 선형 회귀

공부한 시간(x1)	2	4	6	8
과외 수업 횟수(x2)	0	4	2	3
성적(y)	81	93	91	97

$$y = a_1 x_1 + a_2 x_2 + b$$

1-2. 다중 선형 회귀에서 경사하강법

다중 선형 회귀



1-2. 다중 선형 회귀에서 경사하강법

$$a\text{로 편미분 한 결과} = \frac{2}{n}\sum(ax_i + b - y_i)x_i$$

$$b\text{로 편미분 한 결과} = \frac{2}{n}\sum(ax_i + b - y_i)$$

$a_1 := a_1 -$ 학습률 $\cdot$ a_1로 편미분한 결과

$a_2 := a_2 -$ 학습률 $\cdot$ a_2로 편미분한 결과

$b := b -$ 학습률 $\cdot$ b로 편미분한 결과

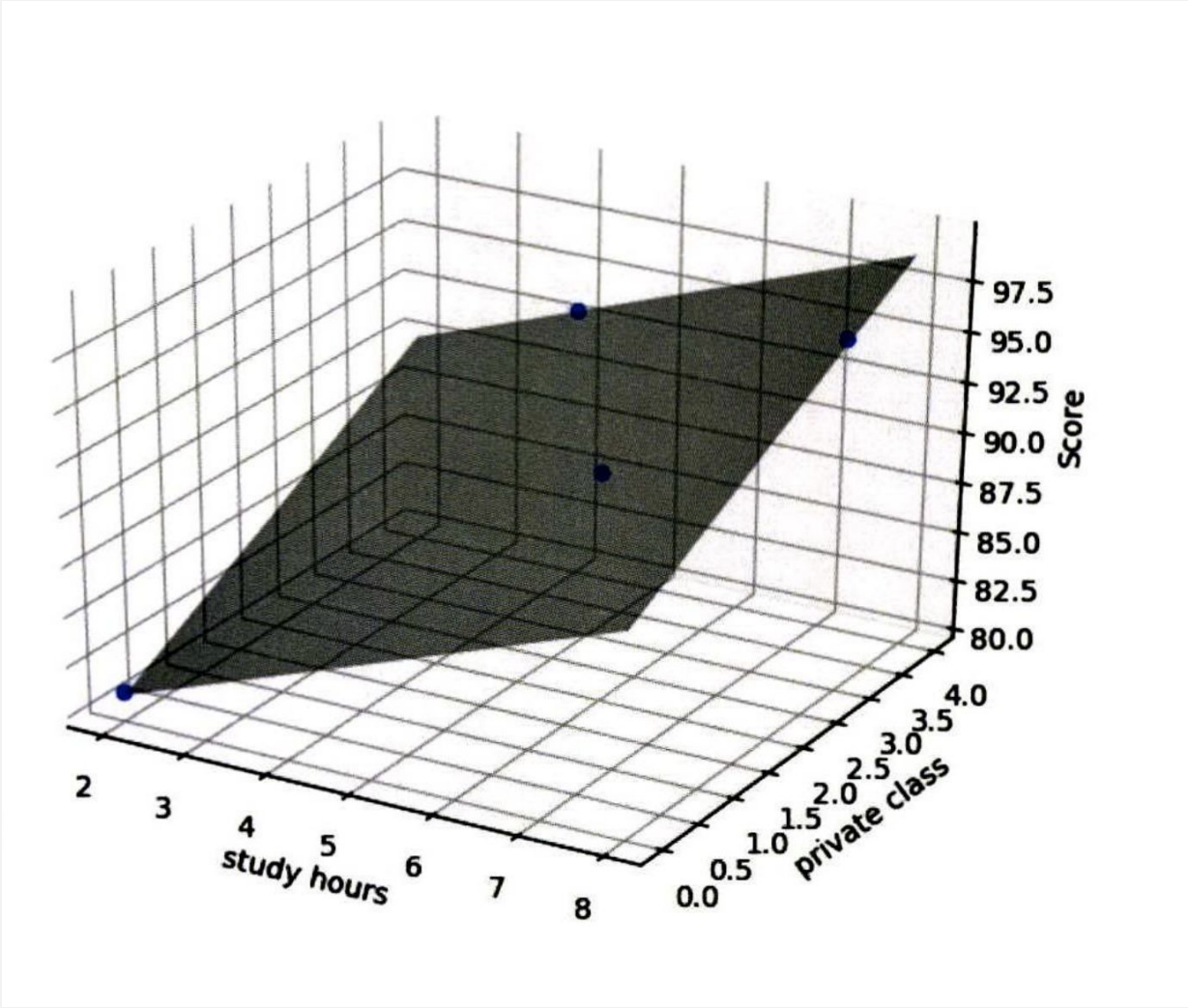
1-2. 다중 선형 회귀에서 경사하강법

다중 선형 회귀의 최종 목표

오차의 값이 최소가 되도록 하는 a1, a2, b를 찾는 것

1-2. 다중 선형 회귀에서 경사하강법

예측 평면



다중 선형 회귀 ver.
경사하강법 실습 타임!

🎈 🎈 쉬는 시간! 🎈 🎈

3.
pre-class quiz 해설

3. pre-class quiz 해설

Q1. 다음 중 편미분 결과가 올바르지 않은 것은?

a.

$z=f(x,y,r)=xy+yr+rx$ 일 때,

$\frac{\partial z}{\partial x}=y+r,\quad \frac{\partial z}{\partial y}=x+r,\quad \frac{\partial z}{\partial r}=y+x$

⭕

b. ✅

$f(x,y)=x^2+y^2+xy$ 일 때,

x에 대해 편미분 하면 $f_x(x,y)=2y+x$

❌

c.

$\frac{1}{n}\sum(y_i-(ax_i+b))^2$ 일 때,

a로 편미분 한 결과 $=\frac{2}{n}\sum(ax_i+b-y_i)x_i$

⭕

편미분이란 x,y,z 같이 여러 변수가 식 안에 있을 때,
모든 변수를 미분하는 것이 아니라 우리가 원하는 한 가지 변수만 미분하고 그 외에는 모두 상수 취급하는 것
👉 2x+y

3. pre-class quiz 해설

Q2. 다음 문장의 빈칸에 들어갈 알맞은 말은?

경사하강법은 마치 안개가 낀 산을 가장 경사가 가파른 길을 찾아 여러 걸음 내딛으면서 높이가 가장 낮은 곳을 찾는 것과 같다.
이때, 경사하강법에서 걸음의 보폭을 (a)라고 할 수 있다.

즉, 경사하강법이란 오차의 변화에 따라 (b) 그래프를 만들고 적절한 (a) 을(를) 설정해 미분 값이 (c)인 지점을 구하는 것이다.

(a) 학습률
(b) 이차 함수
(c) 0

학습률의 크기에 따라 오차의 최소점에 수렴하는 속도가 달라진다.

학습률이 너무 크다면 오차 최소점으로 내려가지 못하고 치솟아 버리고,
반대로 학습률 너무 작다면 오차 최소점으로 내려가는 속도가 느려진다.

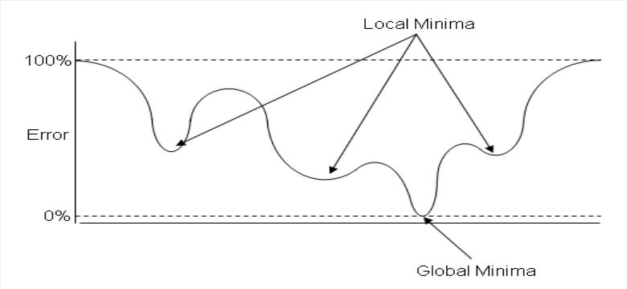
3. pre-class quiz 해설

Q3. 경사하강법의 장점으로 옳은 것을 선택하세요.

1. 지역 최소점(local minima)에 빠질 수도 있다.
2. 안장점(saddle point)에서 벗어나지 못한다.
3. 전체 데이터를 한 번에 학습시키기 때문에 데이터 양이 많으면 학습이 느려진다.

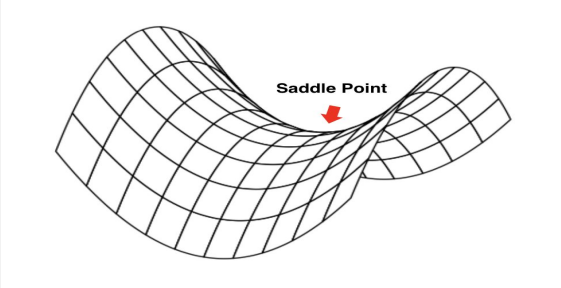
✅ 4. 딥러닝에서 사용되는 복잡한 오차 함수의 최적화를 효율적으로 할 수 있다.

1.



전역 최소점(global minima)이 아니라 가짜 최소점인 지역 최소점(local minima)에 빠져 실제 최소점이 전역 최소점에 도달하지 못할 수 도 있다.

2.



안장점은 최소점은 아니지만 기울기의 값이 0이 나오는 지점이다.

3.

경사하강법은 한 번에 모든 데이터를 넣어 학습시키기 때문에 한 번 데이터를 학습시킬 때 많은 메모리를 사용하여 큰 데이터에 적합하지 않다.

👉 **확률적 경사 하강법(SGD)**
확률적 경사 하강법은 모든 데이터를 넣는 것이 아니라 데이터를 조금씩 분할해서 넣어 큰 데이터라도 학습시킬 수 있는 경사 하강법이다.
(모딥 교재 p. 123)

4.

1차 함수, 2차 함수와 같이 단순한 함수는 직접 미분계수를 구하기 쉽지만 딥러닝에서 사용되는 오차 함수들은 직접 미분계수를 구하는 것이 어렵다.

하지만 **컴퓨터로 미분계수를 경사하강법으로 구하는 것은 간단하고 효율적**이기 때문에 딥러닝에서는 경사하강법을 최적화를 위해 사용하고 있다.

3. pre-class quiz 해설

Q4. 다음은 학습률을 달리하여 경사하강법을 적용한 결과이다. 어떤 것이 가장 잘 최적화 된 것인가?

```
epoch=0, a편미분=-928.0000, b편미분=-181.0000
epoch=100, a편미분=3.0028, b편미분=-17.9197
epoch=200, a편미분=2.1701, b편미분=-12.9500
epoch=300, a편미분=1.5682, b편미분=-9.3586
epoch=400, a편미분=1.1333, b편미분=-6.7632
epoch=500, a편미분=0.8190, b편미분=-4.8875
epoch=600, a편미분=0.5919, b편미분=-3.5321
epoch=700, a편미분=0.4277, b편미분=-2.5525
epoch=800, a편미분=0.3091, b편미분=-1.8446
epoch=900, a편미분=0.2234, b편미분=-1.3331
epoch=1000, a편미분=0.1614, b편미분=-0.9634
epoch=1100, a편미분=0.1167, b편미분=-0.6962
epoch=1200, a편미분=0.0843, b편미분=-0.5031
epoch=1300, a편미분=0.0609, b편미분=-0.3636
epoch=1400, a편미분=0.0440, b편미분=-0.2628
epoch=1500, a편미분=0.0318, b편미분=-0.1899
epoch=1600, a편미분=0.0230, b편미분=-0.1372
epoch=1700, a편미분=0.0166, b편미분=-0.0992
epoch=1800, a편미분=0.0120, b편미분=-0.0717
epoch=1900, a편미분=0.0087, b편미분=-0.0518
epoch=2000, a편미분=0.0063, b편미분=-0.0374
```

a.

학습률이 너무 낮음

```
epoch=0, a편미분=-928.000000, b편미분=-181.000000
epoch=100, a편미분=2.167772, b편미분=-12.936335
epoch=200, a편미분=1.130923, b편미분=-6.748865
epoch=300, a편미분=0.590001, b편미분=-3.520872
epoch=400, a편미분=0.307802, b편미분=-1.836833
epoch=500, a편미분=0.160580, b편미분=-0.958273
epoch=600, a편미분=0.083774, b편미분=-0.499929
epoch=700, a편미분=0.043705, b편미분=-0.260812
epoch=800, a편미분=0.022801, b편미분=-0.136065
epoch=900, a편미분=0.011895, b편미분=-0.070985
epoch=1000, a편미분=0.006206, b편미분=-0.037033
epoch=1100, a편미분=0.003237, b편미분=-0.019320
epoch=1200, a편미분=0.001689, b편미분=-0.010079
epoch=1300, a편미분=0.000881, b편미분=-0.005258
epoch=1400, a편미분=0.000460, b편미분=-0.002743
epoch=1500, a편미분=0.000240, b편미분=-0.001431
epoch=1600, a편미분=0.000125, b편미분=-0.000747
epoch=1700, a편미분=0.000065, b편미분=-0.000390
epoch=1800, a편미분=0.000034, b편미분=-0.000203
epoch=1900, a편미분=0.000018, b편미분=-0.000106
epoch=2000, a편미분=0.000009, b편미분=-0.000055
```

b.

학습률이 너무 낮음

```
epoch=0, a편미분=-928.000000, b편미분=-181.000000
epoch=100, a편미분=1.563174, b편미분=-9.328872
epoch=200, a편미분=0.588122, b편미분=-3.509662
epoch=300, a편미분=0.221261, b편미분=-1.320389
epoch=400, a편미분=0.083242, b편미분=-0.496751
epoch=500, a편미분=0.031317, b편미분=-0.186885
epoch=600, a편미분=0.011782, b편미분=-0.070309
epoch=700, a편미분=0.004433, b편미분=-0.026451
epoch=800, a편미분=0.001668, b편미분=-0.009951
epoch=900, a편미분=0.000627, b편미분=-0.003744
epoch=1000, a편미분=0.000236, b편미분=-0.001409
epoch=1100, a편미분=0.000089, b편미분=-0.000530
epoch=1200, a편미분=0.000033, b편미분=-0.000199
epoch=1300, a편미분=0.000013, b편미분=-0.000075
epoch=1400, a편미분=0.000005, b편미분=-0.000028
epoch=1500, a편미분=0.000002, b편미분=-0.000011
epoch=1600, a편미분=0.000001, b편미분=-0.000004
epoch=1700, a편미분=0.000000, b편미분=-0.000002
epoch=1800, a편미분=0.000000, b편미분=-0.000001
epoch=1900, a편미분=0.000000, b편미분=-0.000000
epoch=2000, a편미분=0.000000, b편미분=-0.000000
```

c. ✅

학습률이 적절

```
epoch=0, a편미분=-928.000000, b편미분=-181.000000
epoch=100, a편미분=-410540661…, b편미분=-687952465…
epoch=200, a편미분=-1808106508725979524099 1…
epoch=300, a편미분=-7963278347508860372119 6…
epoch=400, a편미분=-3507193947583643199128 5…
epoch=500, a편미분=-1544641396318292857053 5…
epoch=600, a편미분=-6802923017314043431097 9…
epoch=700, a편미분=-2996149247962073603238 6…
epoch=800, a편미분=-1319566647045195460908 3…
epoch=900, a편미분=-5811646856973047246170 2…
epoch=1000, a편미분=-2559570542783503426052…
epoch=1100, a편미분=-1127288275546958526726…
epoch=1200, a편미분=-4964812787709603590602…
epoch=1300, a편미분=-2186607148472732514469…
epoch=1400, a편미분=-9630274143645525885373…
epoch=1500, a편미분=-4241373680066163896540…
epoch=1600, a편미분=-1867989470043081069702…
epoch=1700, a편미분=-8227015404446497623694…
epoch=1800, a편미분=-3623349250648555523953…
epoch=1900, a편미분=nan, b편미분=nan
epoch=2000, a편미분=nan, b편미분=nan
```

d.

학습률이 너무 큼

SMARCLE

3. pre-class quiz 해설

Q5. Q4의 d번에서는 nan이라는 결측치가 출력되고 있다. 이처럼 nan이 출력되는 상황에서 이를 고치기 위해 시도해볼 수 있는 방법으로 어떤 것이 있는가? 정답을 모두 골라라.

1. learning rate를 키운다.
2. ✅ learning rate를 줄인다.
3. 에포크 값을 키운다.
4. ✅ 데이터 값 사이의 차이를 줄인다.(데이터 전처리)

1.

learning rate를 키우면 오히려 더 발산한다.

2.

앞 문제의 4번의 경우가 learning rate가 너무 커서 overshooting이 발생하는 경우이다.

이때 learning rate 값을 줄이면 overshooting을 막을 수 있다.

3.

에포크는 얼마나 반복하는지 나타내는 값이다. 에포크를 키운다고 해결되지 않는다.

learning rate가 너무 작아서 채 수렴하지 못한 경우에는 에포크 값을 키워서 해결할 수 있다.

4.

train하는 데이터의 스펙트럼이 너무 넓으면(두 개의 독립변수 값이 너무 차이가 크면) cost function도 넓은 그릇 같은 모양처럼 그려진다.
때문에 경사하강법 알고리즘이 수렴하는데 오래 걸리거나 그래프 바깥으로 튀어나가 버릴 수가 있다.

이럴 때 Normalize를 통해 데이터를 전처리 하고 스케일을 조정하면 이를 막을 수 있다. (normalize도 다양한 방법이 있다.)

예제 문제

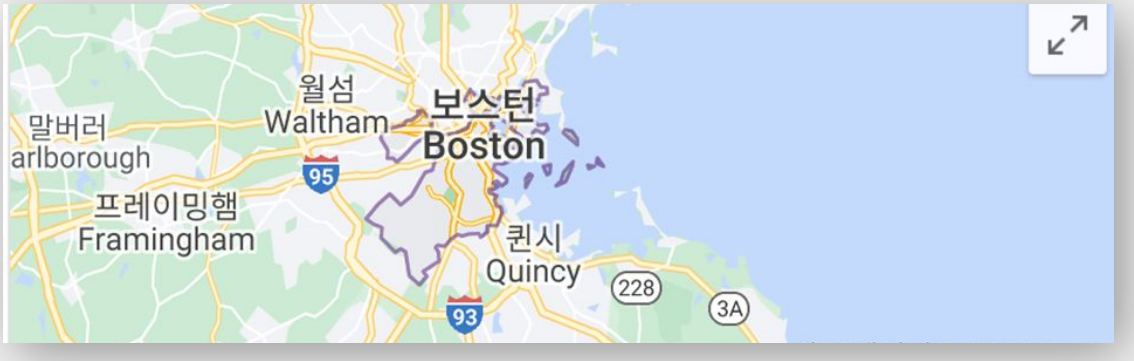
실습 타임!

4. 예제 문제

〈 미국 Boston의 집 값 예측 〉

✔ 문제!

미국 Boston의 집 값은
다음 중 어떤 요인에 더 영향을 많이 받을까요?
다중 선형 회귀로 예측해봅시다!



A 거주할 수 있는 방 개수 & 지역의 교사와 학생 수 비율 **VS.** 지역별 범죄 발생률 & 일산화질소 농도 **B**

✔ 힌트!

※ sklearn 모듈을 사용하기 위해 실습은 코랩에서 진행해주세요!

1) scikit-learn 내장 데이터셋의 데이터를 이용!

```
from sklearn.datasets import load_boston

bst = load_boston()
bstDF = pd.DataFrame(bst.data, columns = bst.feature_names)
```

2) 독립변수로 'RM'과 'PTRATIO' 갖는 모델 하나, 'CRIM'과 'NOX' 갖는 모델 하나를 설정!

```
bstDF1 = bstDF[['RM', 'PTRATIO']] # 거주할 수 있는 방 개수, 지역의 교사와 학생 수 비율
bstDF2 = bstDF[['CRIM', 'NOX']] # 지역별 범죄 발생률, 일산화질소 농도
```