

회귀를 통해 이해하는 Cost Function

DataScience

Exported on 11/12/2020

Table of Contents

1	기본 개념 잡기.....	5
1.1	만약 주택의 넓이과 가격이라는 데이터가 있고 주택가격을 예측한다면	5
1.2	머신러닝 모델 만들기	5
1.3	어떻게 만들까? 그~ 모델	6
1.4	만약 1차 함수라면~	6
1.5	선형 회귀	7
1.6	모델을 구성하는 파라미터를 어떻게 찾을까?	7
1.7	간단한 수학~ 점 세 개를 표현하는 직선	8
1.8	쉬울 수도 있다.....	8
1.9	그렇지 않을 수도 있다	9
1.10	직선상에 있지 않은 세 점을 직선으로 표현하기	9
1.11	시도 1	10
1.12	시도 2	10
1.13	대부분은 이렇게 하겠지.....	11
1.14	근거는?	11
1.15	실제 데이터(점 세 개)와 구해야할 모델(h)	12
1.16	먼저 각각의 에러를 구하고.....	12
1.17	각각의 에러를 제곱하고.....	13
1.18	에러 각각을 제곱하고 평균을 구함	13
1.19	여러분은 지금 그 유명한 Cost Function을 보고 계십니다.....	14
1.20	Cost Fnc을 최소화할 수 있다면 최적의 직선을 찾을 수 있다.....	14
1.21	계산해 볼까요?	15
1.22	J를 최소로 만들 수 있을까요?.....	15
1.23	Python 유저라면 뭐 코드로 승부~.....	16
1.24	그리고 최솟값 찾기	16
1.25	그럼 저 최솟값 지점은 어떻게 구하죠?.....	17
1.26	뭐 손으로?	17
1.27	혹은 Python으로? Sympy install	18

1.28 그리고 Symbolic 연산이 가능하다	18
1.29 Python이든 손으로 직접하든 구할 수 있습니다.	18
1.30 애초 감으로 그린 것과는 조금 차이가 있네요.....	19
2 Cost Function	20
2.1 Cost Function - 데이터와 모델이 완전 일치하면?	20
2.2 Cost Function - 조금 빗나가면.....	20
2.3 Cost Function - 더 빗나가면	21
2.4 그러나 실제는 데이터는 너무 복잡해서 손으로 풀기 어렵다.....	21
2.5 Cost Function의 최솟값을 찾기 위해서 특단의 대책이 필요.....	22
2.6 How?	22
3 Gradient Descent.....	23
3.1 랜덤하게 임의의 점 선택	23
3.2 임의의 점에서 미분(or 편미분)값을 계산해서 업데이트	23
3.3 목표점의 오른쪽이라면.....	24
3.4 목표점의 왼쪽이었다면.....	24
3.5 여기서 또 중요한 개념 학습률 - Learning Rate.....	25
3.6 학습률이 작다면	25
3.7 학습률이 크다면	26
4 다변수 데이터에 대한 회귀.....	27
4.1 여러개의 특성(feature)	27
4.2 행렬식으로 표현	27
5 간단한 예제 하나 - Boston 집값 예측	28
5.1 보스톤 집 가격 데이터	28
5.2 데이터 읽기.....	28
5.3 각 특성의 의미.....	29
5.4 데이터 파악을 위해 pandas로 정리	29
5.5 Price에 대한 histogram.....	29
5.6 집값에 대한 히스토그램.....	30
5.7 각 특성별 상관계수 확인	30
5.8 상관계수 조사 결과	31

5.9 RM과 LSTAT와 PRICE의 관계에 대해 좀 더 관찰해보자.....	31
5.10 저소득층 인구가 낮을 수록, 방의 갯수가 많을 수록 집 값이 높아짐???	32
5.11 일단 데이터를 나누고	32
5.12 LinearRegression을 사용하고.....	32
5.13 모델 평가는 RMS로.....	33
5.14 성능 확인	33
5.15 결과.....	33
5.16 그런데, LSTAT를 사용하는 것이 맞는 걸까?	34
5.17 당연히 성능은 나빠진다.....	34
5.18 이런 류의 고민은 아마 계속될 것~	35

1 기본 개념 잡기

1.1 만약 주택의 넓이과 가격이라는 데이터가 있고 주택가격을 예측한다면

• 머신러닝 모델을 어떻게 만들까요?

주택 가격 예측

학습 데이터셋 (Training Dataset)

주택 규모(m^2)	주택 가격(백만원)
220	325
135	295
85	250
55	176
...	...

“ 학습 데이터 각각에 정답(주택 가격)이 주어져 있으므로 **지도학습**(Supervised Learning)이며, 주택 가격을 연속된 값으로 예측하는 것이므로 **회귀**(Regression) 문제임 ”

1.2 머신러닝 모델 만들기

• 머신러닝 모델을 어떻게 만들까요?

```

graph TD
    A[학습 데이터셋] --> B[학습 알고리즘]
    B --> C[" $h$ "]
    D[주택 크기] --> C
    C --> E["주택 가격  
(예측값)"]
    
```

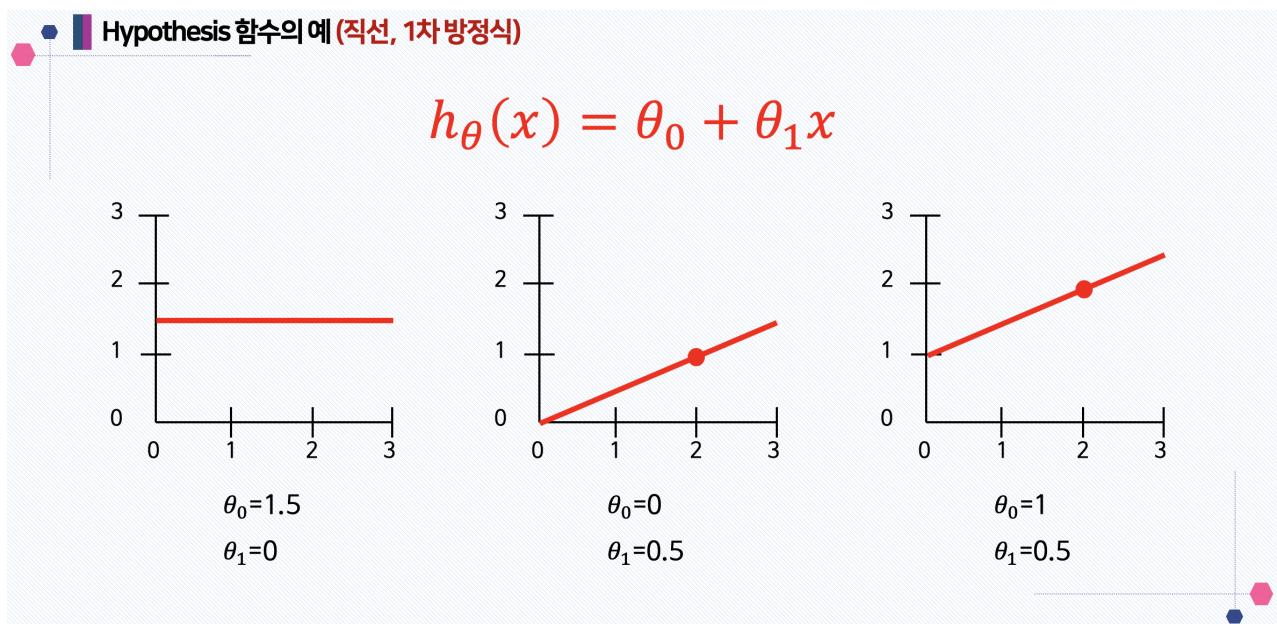
? Hypothesis (가설 = 모델) h 를 어떻게 만들면 좋을까요?

1.3 어떻게 만들까? 그~ 모델

• Hypothesis h 를 어떻게 만들까요?

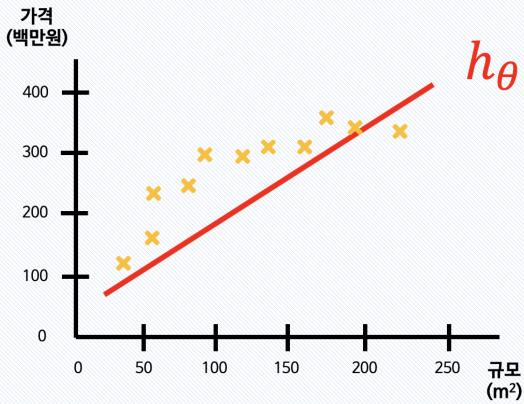
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

1.4 만약 1차 함수라면~



1.5 선형 회귀

• 선형 회귀 (Linear Regression) 문제의 정의



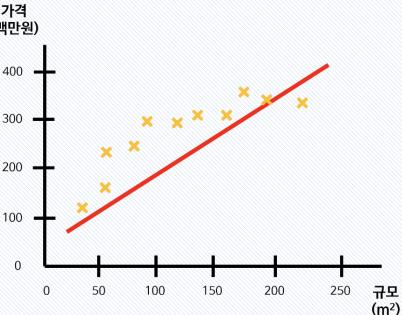
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

“ 입력 변수(특징) x 가 하나인 경우, 선형 회귀 (Linear Regression) 문제는 주어진 학습데이터와 가장 잘 맞는 Hypothesis 함수 h 를 찾는 문제가 됨 ”

1.6 모델을 구성하는 파라미터를 어떻게 찾을까?

• Hypothesis h 의 파라미터 θ_0 와 θ_1 을 어떻게 찾을까요?

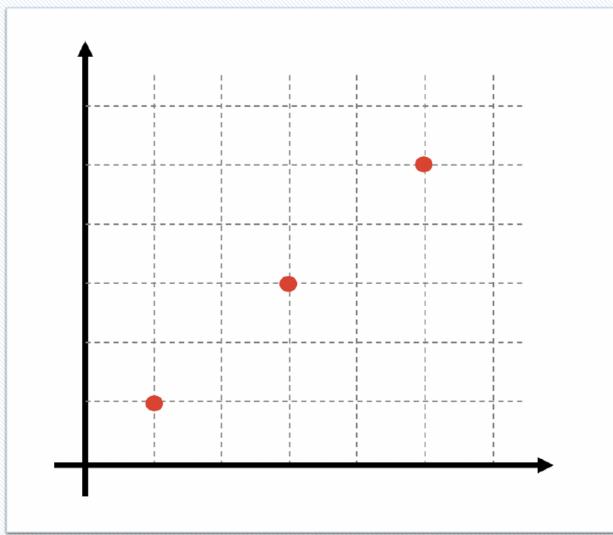
주택 규모(m^2)	주택 가격(백만원)
$x^{(1)} = 220$	$y^{(1)} = 325$
$x^{(2)} = 135$	$y^{(2)} = 295$
$x^{(3)} = 85$	$y^{(3)} = 250$
$x^{(4)} = 55$	$y^{(4)} = 176$
...	...



? 주어진 학습데이터 $x^{(i)}$ 에 대해 정답 $y^{(i)}$ 와 예측값 $h(x^{(i)})$ 의 차이가 최소가 되게 파라미터 θ_0 와 θ_1 의 값을 결정하면 어떨까요?

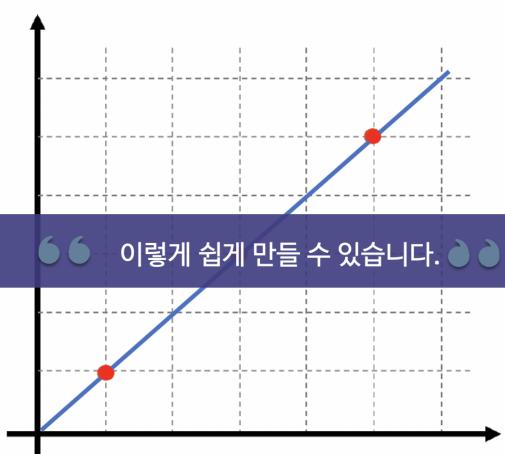
1.7 간단한 수학~점 세 개를 표현하는 직선

만약 점 세 개가 있고, 이 점을 하나의 직선으로 표현한다면?



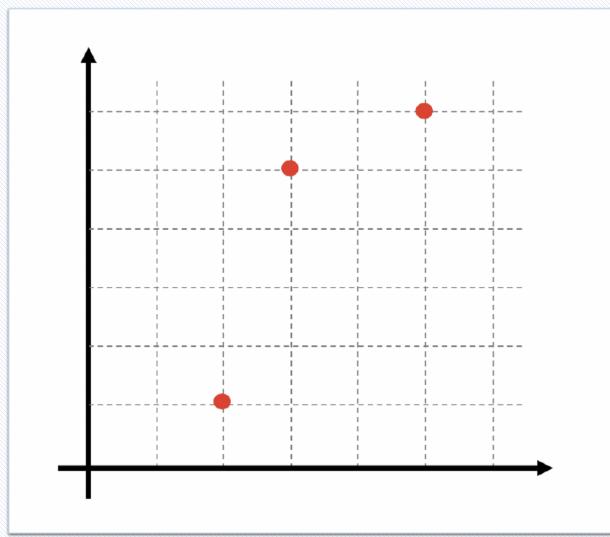
1.8 쉬울 수도 있다

만약 점 세 개가 있고, 이 점을 하나의 직선으로 표현한다면?



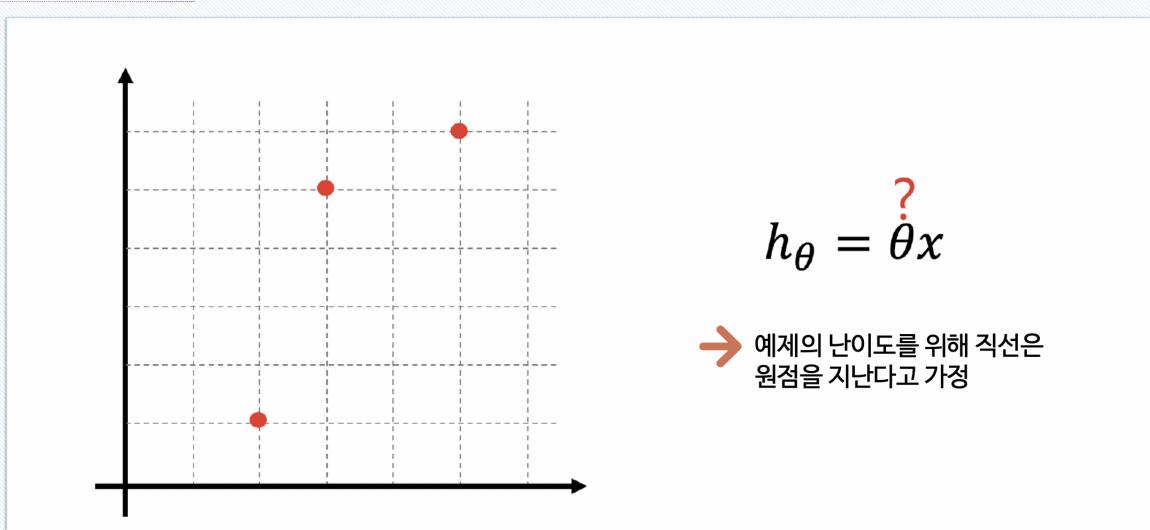
1.9 그렇지 않을 수도 있다

• 그런데 점이 이렇게 되어 있다면?

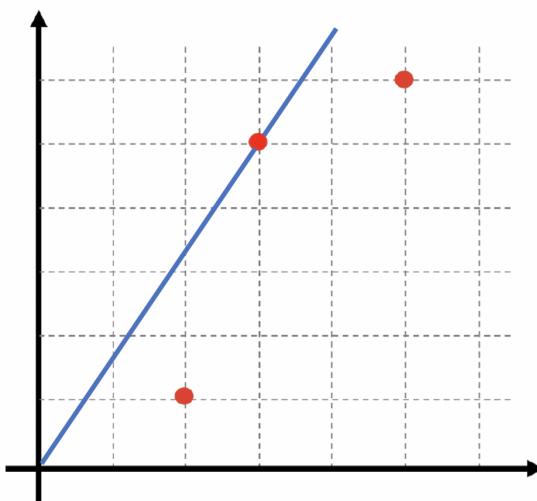


1.10 직선상에 있지 않은 세 점을 직선으로 표현하기

• 이 점 세 개를 하나의 직선으로 표현할 수 있을까요?



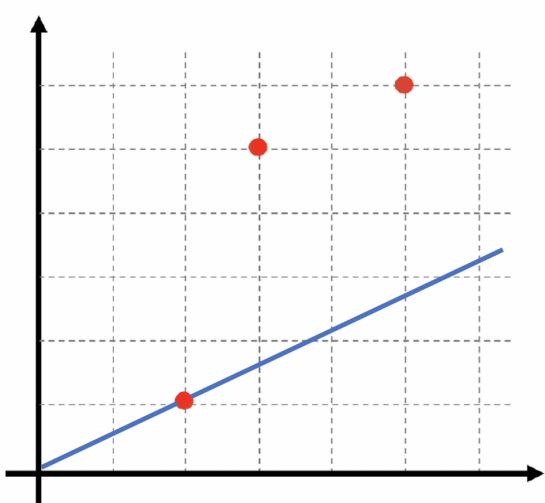
1.11 시도 1



$$h_{\theta} = \theta x$$

“ 이렇게 구할까요? 🌟🌟

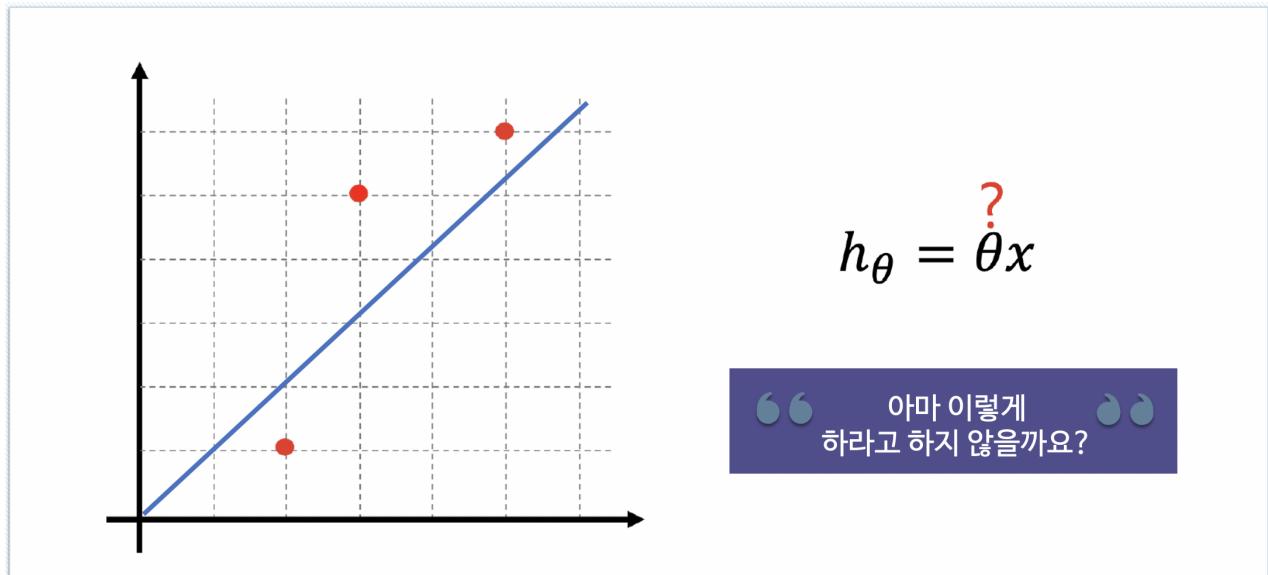
1.12 시도 2



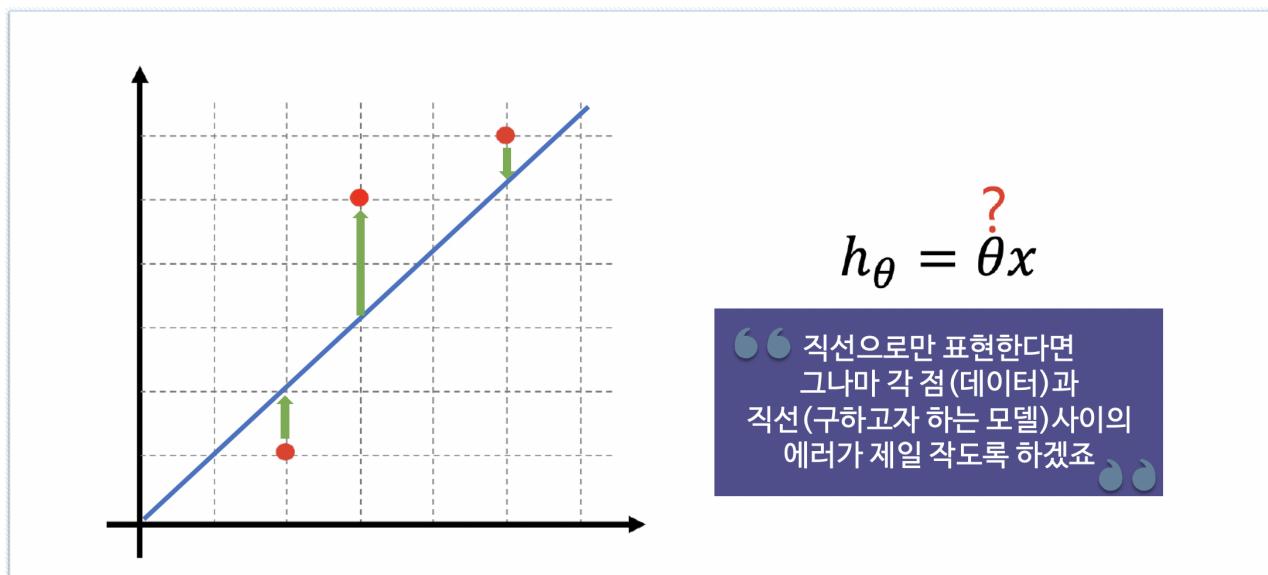
$$h_{\theta} = \theta x$$

“ 또는 이렇게 구할까요? 🌟🌟

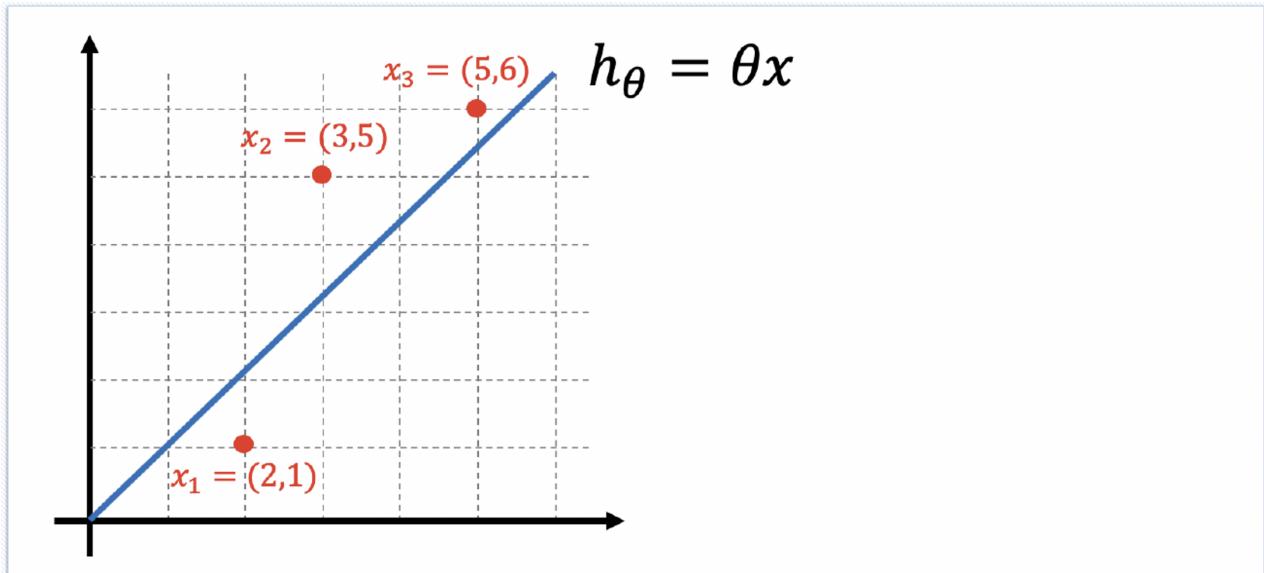
1.13 대부분은 이렇게 하겠지



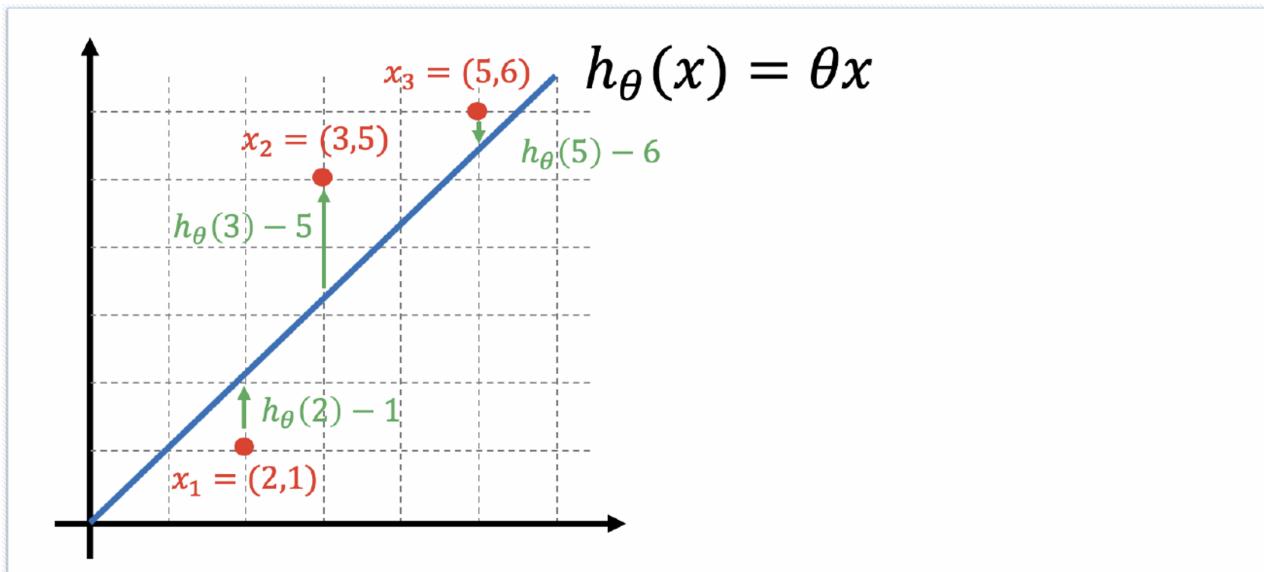
1.14 근거는?



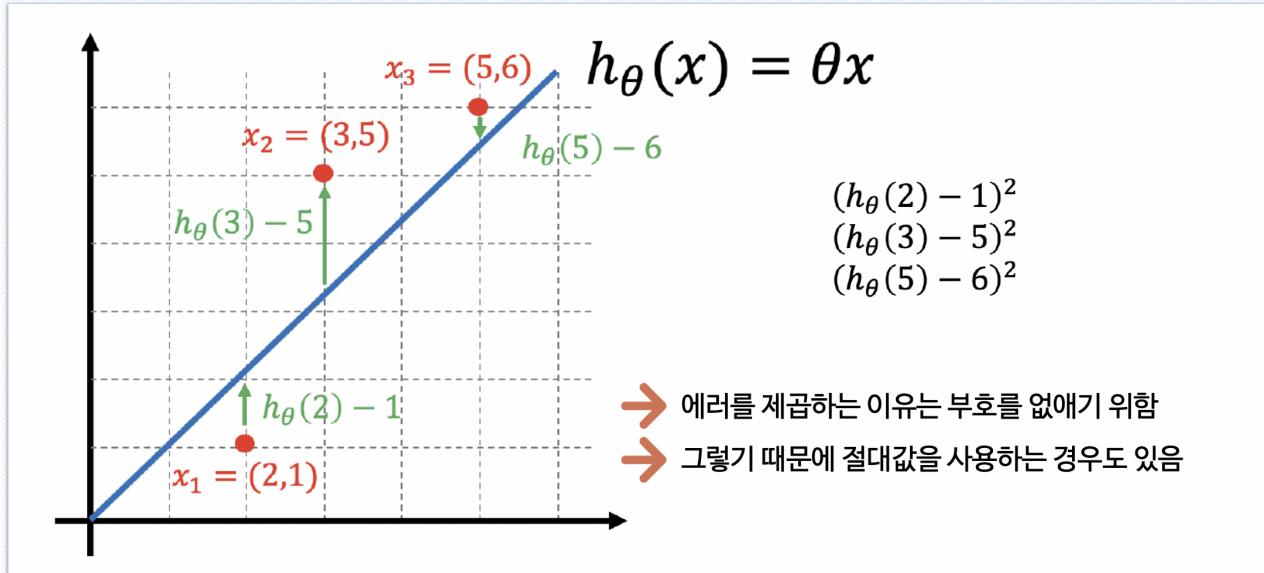
1.15 실제 데이터(점 세 개)와 구해야할 모델(h)



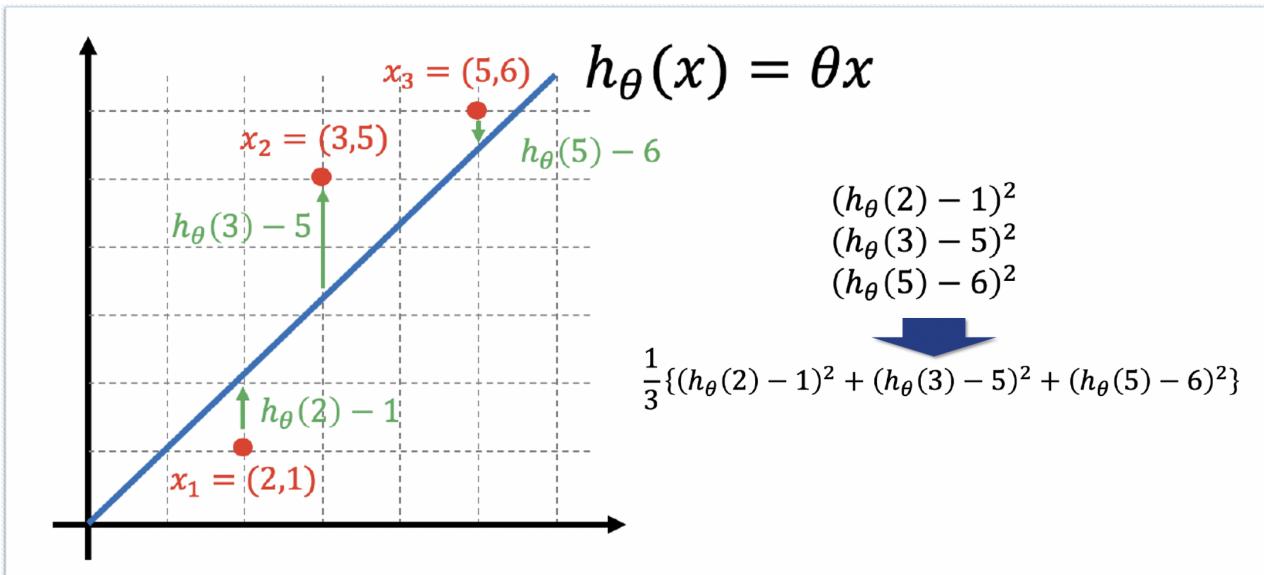
1.16 먼저 각각의 에러를 구하고



1.17 각각의 에러를 제곱하고



1.18 에러 각각을 제곱하고 평균을 구함



1.19 여러분은 지금 그 유명한 Cost Function을 보고 계십니다.

$$J(\theta) = \frac{1}{3} \{(h_{\theta}(2) - 1)^2 + (h_{\theta}(3) - 5)^2 + (h_{\theta}(5) - 6)^2\}$$

1.20 Cost Fnc을 최소화할 수 있다면 최적의 직선을 찾을 수 있다

$$J(\theta) = \frac{1}{3} \{(h_{\theta}(2) - 1)^2 + (h_{\theta}(3) - 5)^2 + (h_{\theta}(5) - 6)^2\}$$

$$\min_{\theta} J(\theta)$$



$$h_{\theta}(x) = \theta x$$

1.21 계산해 볼까요?

$$h_{\theta}(x) = \theta x$$
$$J(\theta) = \frac{1}{3} \{(h_{\theta}(2) - 1)^2 + (h_{\theta}(3) - 5)^2 + (h_{\theta}(5) - 6)^2\}$$



$$J(\theta) = \frac{1}{3} \{(2\theta - 1)^2 + (3\theta - 5)^2 + (5\theta - 6)^2\}$$

1.22 J를 최소로 만들 수 있을까요?

$$J(\theta) = \frac{1}{3} \{(2\theta - 1)^2 + (3\theta - 5)^2 + (5\theta - 6)^2\}$$



1.23 Python 유저라면 뭐 코드로 승부~

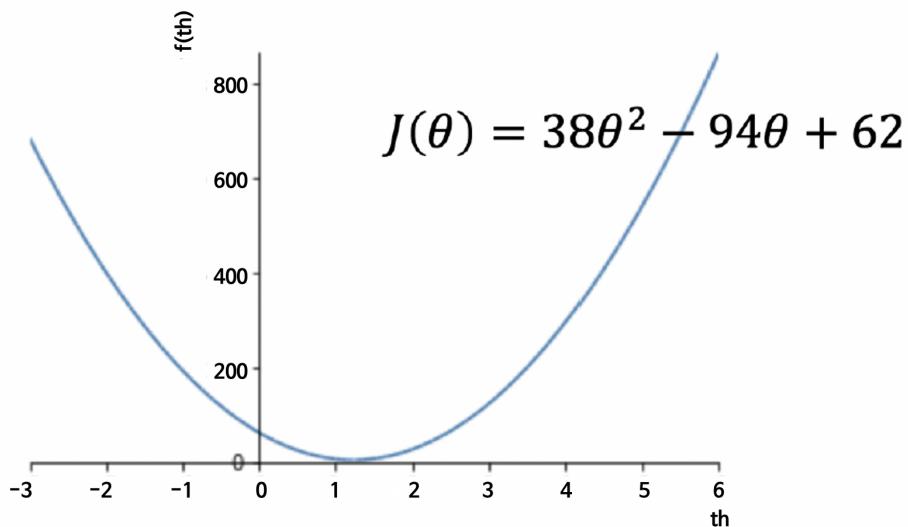
```
In [5]: import numpy as np  
  
np.poly1d([2, -1])**2 + np.poly1d([3, -5])**2 + np.poly1d([5, -6])**2  
Out[5]: poly1d([ 38, -94,  62])
```



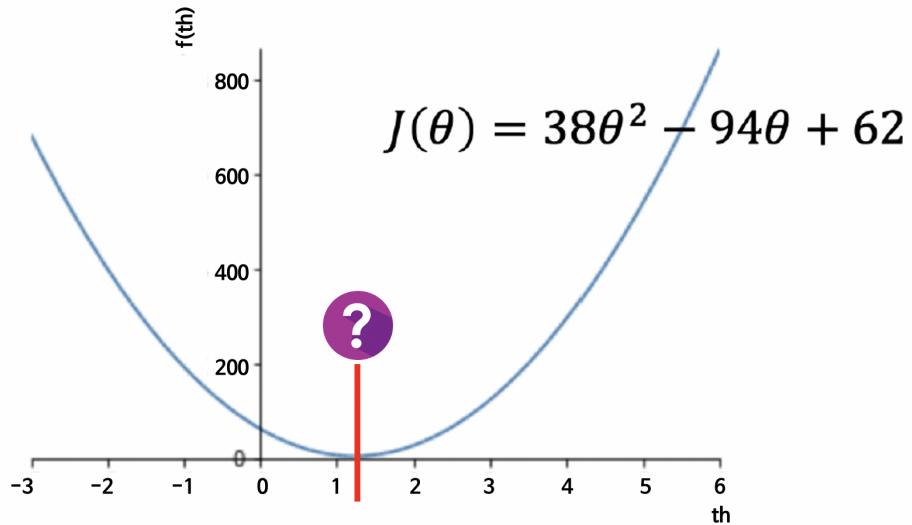
$$J(\theta) = 38\theta^2 - 94\theta + 62$$

→ 숫자의 간결함을 위해 3으로 나누는 것은 하지 않음

1.24 그리고 최솟값 찾기



1.25 그럼 저 최솟값 지점은 어떻게 구하죠?



1.26 뭐 손으로?

$$\begin{aligned}
 J(\theta) &= 38\theta^2 - 94\theta + 62 \\
 \frac{\partial}{\partial \theta} &= 76\theta - 94 \\
 \Rightarrow 76\theta - 94 &= 0 \\
 \theta &= \frac{94}{76} = \boxed{\frac{47}{38}} \approx 1.24
 \end{aligned}$$

1.27 혹은 Python으로? Sympy install

```
!pip install sympy
```

```
Collecting sympy
  Downloading sympy-1.6.2-py3-none-any.whl (5.8 MB)
    |████████████████████████████████| 5.8 MB 118 kB/s eta
0:00:01
  Processing /Users/pw/Library/Caches/pip/wheels/e2/46/78/e78f76c356bca9277
  368f1f97a31b37a8cb937176d9511af31/mpmath-1.1.0-py3-none-any.whl
  Installing collected packages: mpmath, sympy
  Successfully installed mpmath-1.1.0 sympy-1.6.2
```

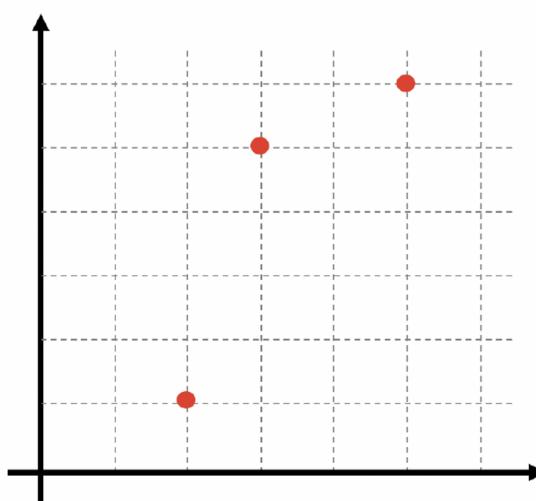
1.28 그리고 Symbolic 연산이 가능하다

```
import sympy as sym

th = sym.Symbol('th')
diff_th = sym.diff(38*th**2 - 94*th + 62, th)
diff_th
```

$76th - 94$

1.29 Python이든 손으로 직접하든 구할 수 있습니다.

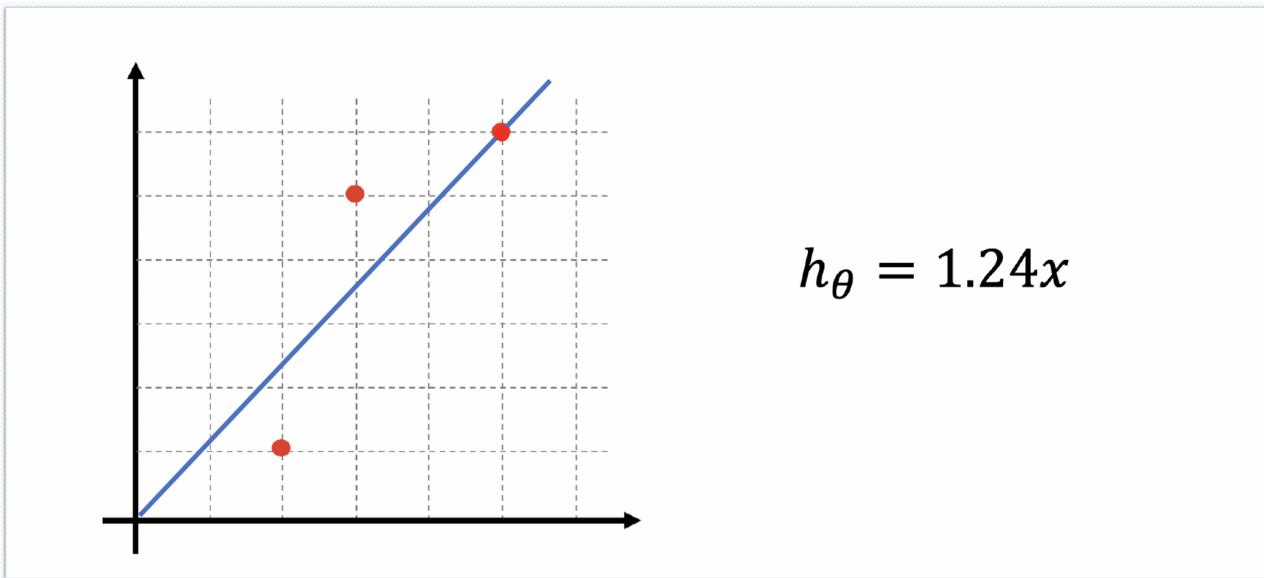


$$h_{\theta}(x) = \theta x$$



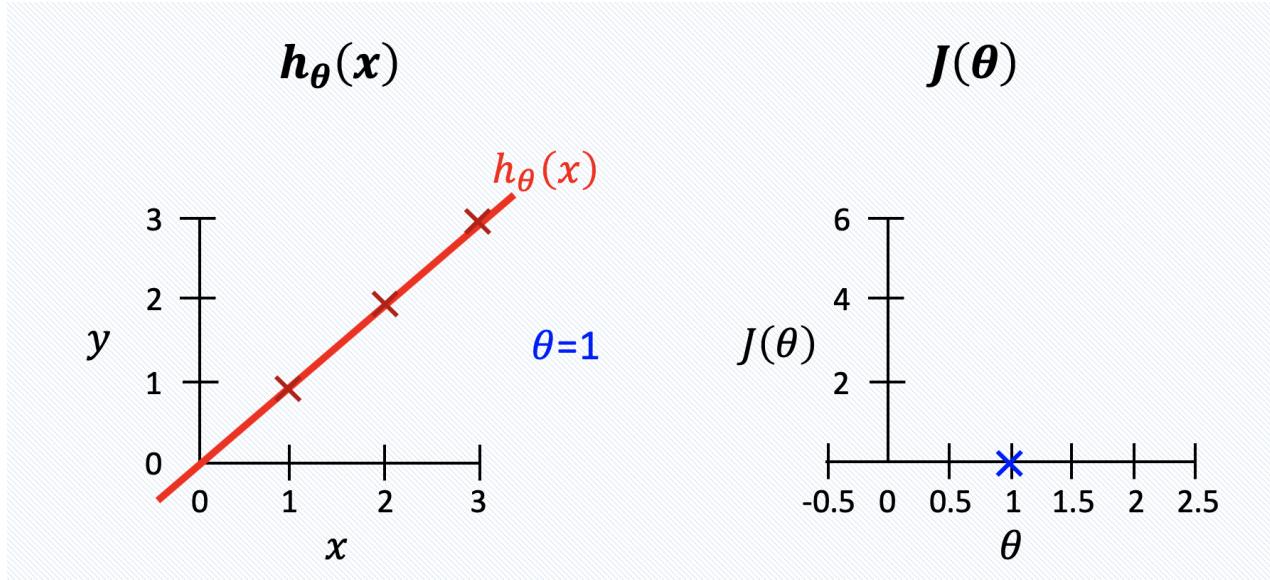
$$h_{\theta}(x) = 1.24x$$

1.30 애초 감으로 그린 것과는 조금 차이가 있네요

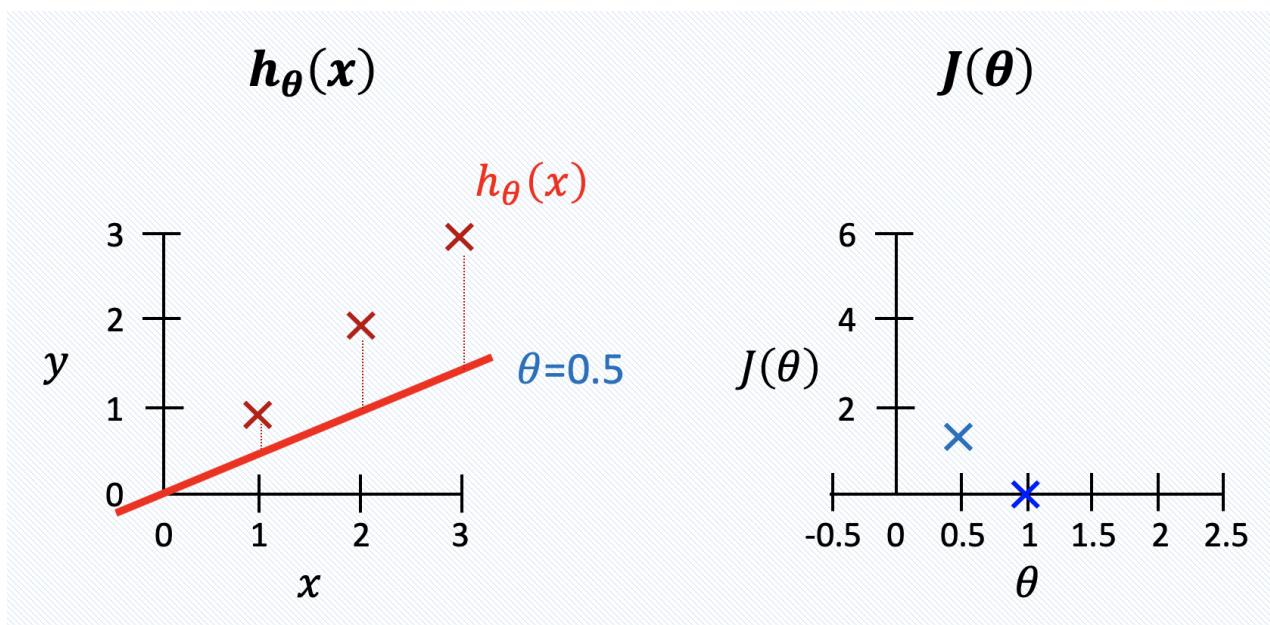


2 Cost Function

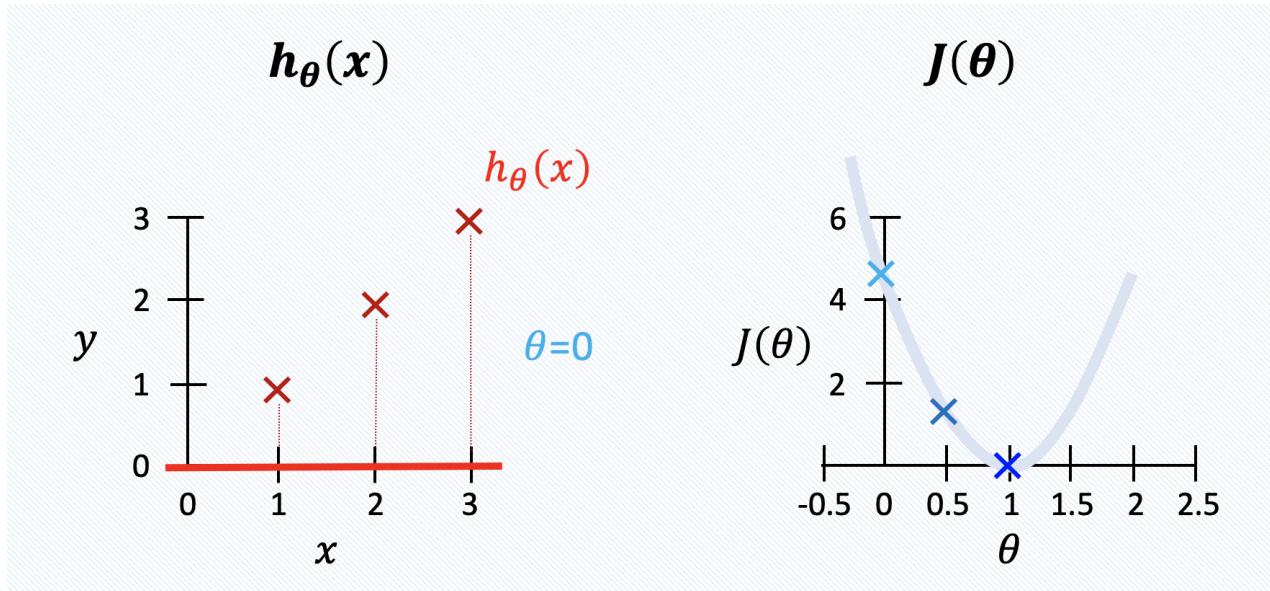
2.1 Cost Function - 데이터와 모델이 완전 일치하면?



2.2 Cost Function - 조금 빗나가면



2.3 Cost Function - 더 빛나가면



2.4 그러나 실제는 데이터는 너무 복잡해서 손으로 풀기 어렵다

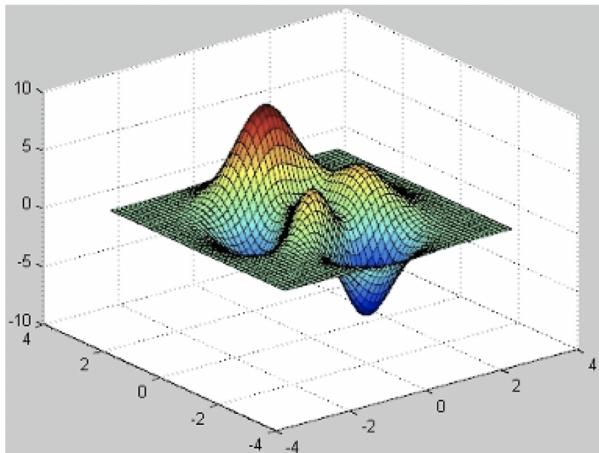
- 그러나 실제 문제는 이렇게 단순하지 않고, 입력 데이터는 여러개일 때가 많습니다.

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33

데이터의 특징 (feature)이 여러개가 존재해서
평면상의 방정식이 아니라 **다차원**에서 고민해야 할 때가 많음

2.5 Cost Function의 최솟값을 찾기 위해서 특단의 대책이 필요

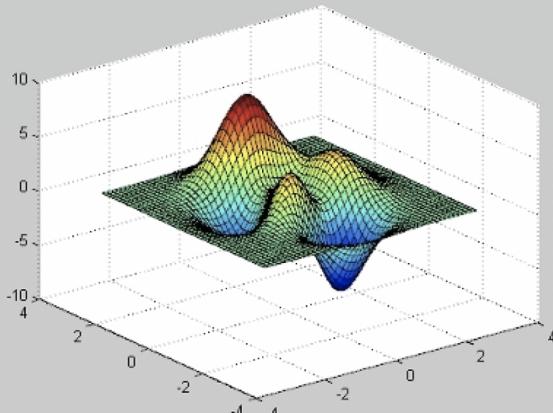
- 우리는 단순하게 한 번의 시도로 COST FUNCTION의 최소값을 구할 수 없을 때가 더 많습니다.



“ 이 정도 데이터면
손으로 문제를 푸는 것은
어려움 ”

2.6 How?

- 우리는 단순하게 한 번의 시도로 COST FUNCTION의 최소값을 구할 수 없을 때가 더 많습니다.

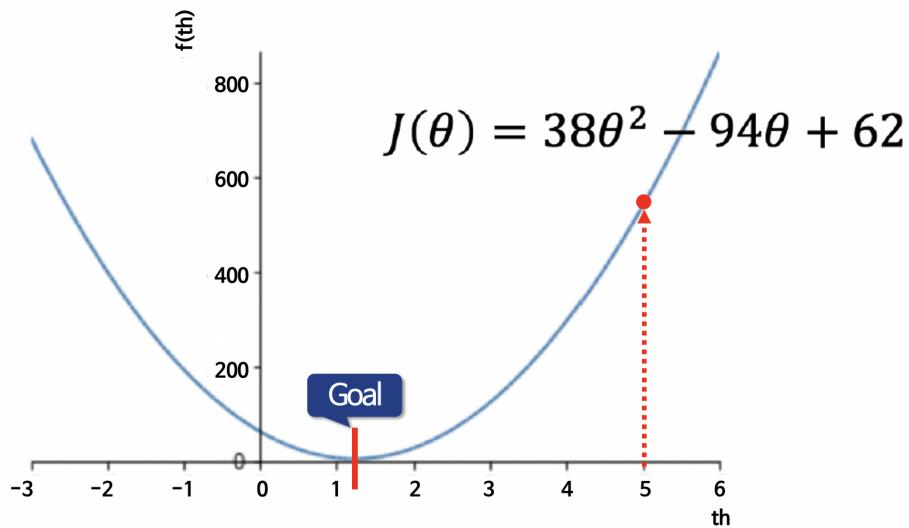


→ 이 정도 데이터면 손으로 문제를 푸는 것은 어려움

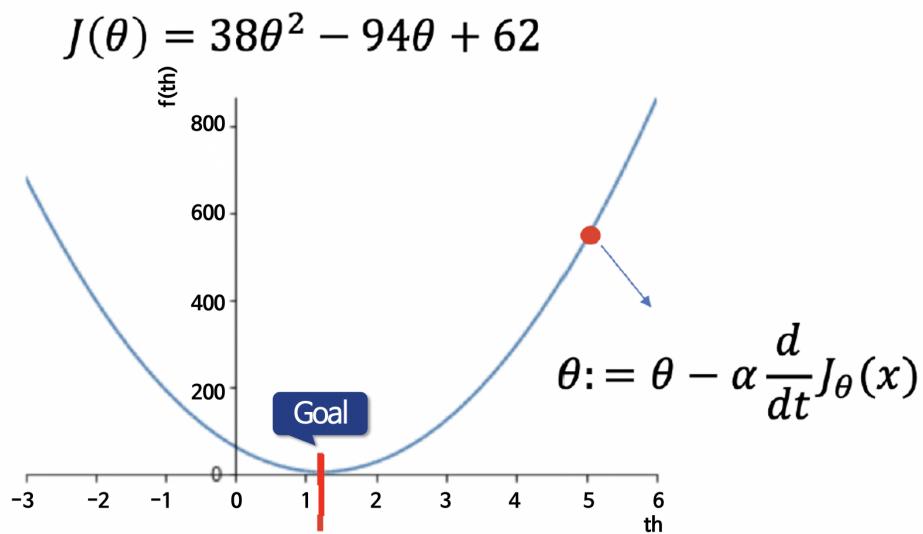
? 그럼 COST FUNCTION이
최소값을 가지는 지점을
어떻게 알 수 있을까요?

3 Gradient Descent

3.1 랜덤하게 임의의 점 선택

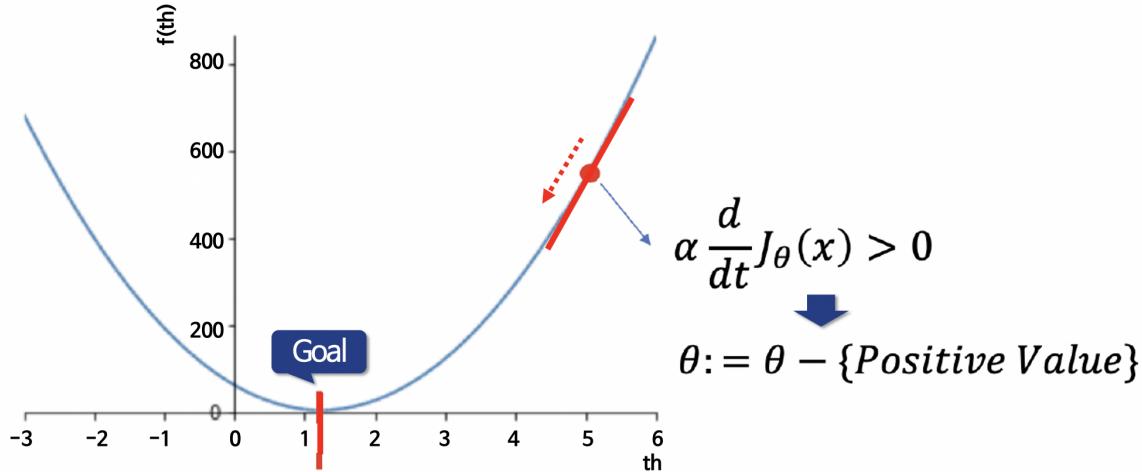


3.2 임의의 점에서 미분(or 편미분)값을 계산해서 업데이트



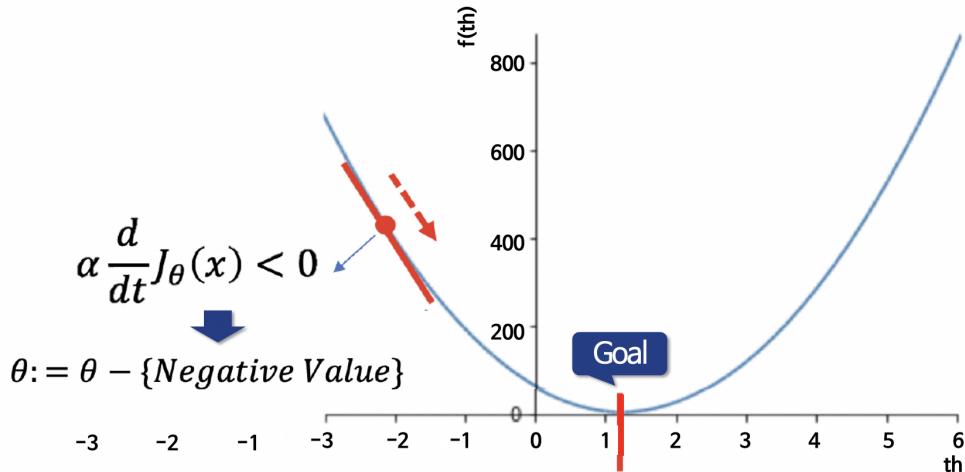
3.3 목표점의 오른쪽이라면

$$J(\theta) = 38\theta^2 - 94\theta + 62$$



3.4 목표점의 왼쪽이었다면

$$J(\theta) = 38\theta^2 - 94\theta + 62$$



3.5 여기서 또 중요한 개념 학습률 - Learning Rate



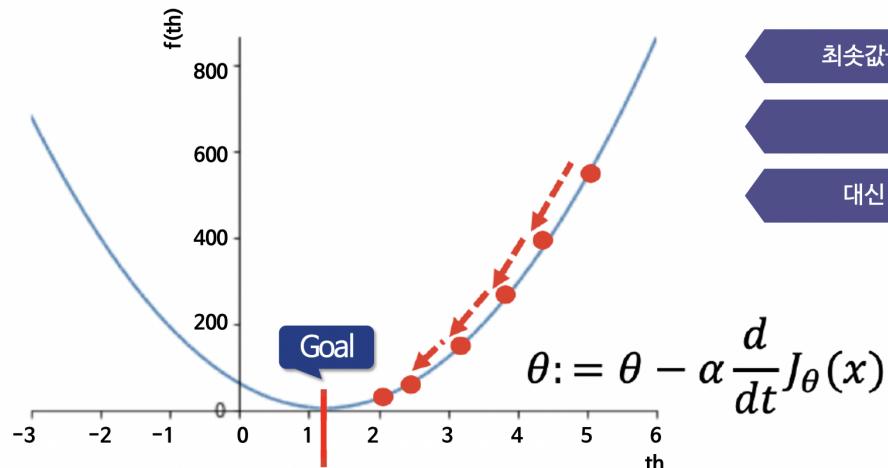
학습률(learning rate)이란?

$$\theta := \theta - \alpha \frac{d}{dt} J_{\theta}(x)$$

→ 위 식에서 alpha는 얼마만큼 theta를 갱신할 것인지를 설정하는 값

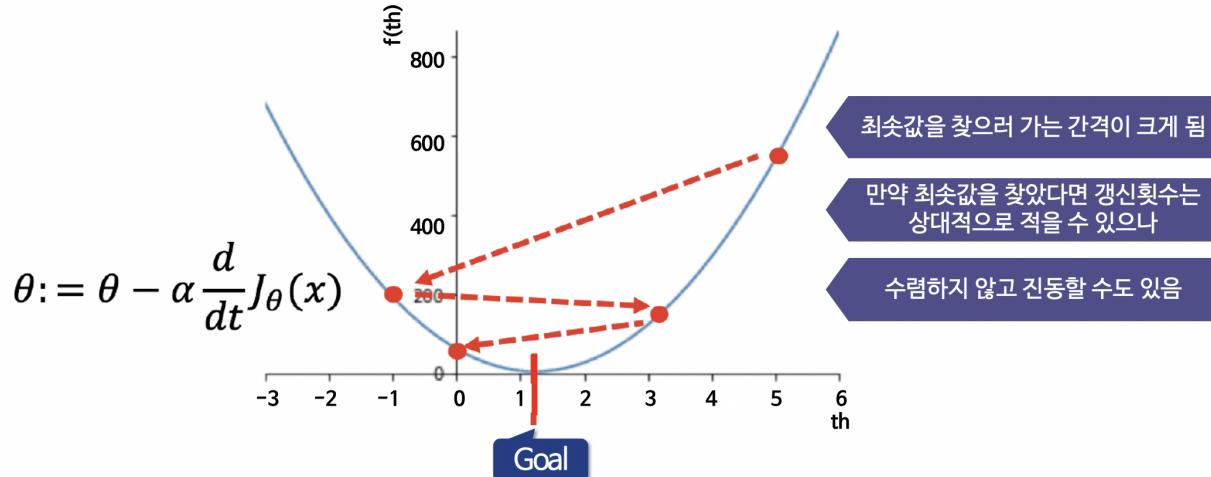
3.6 학습률이 작다면

$$J(\theta) = 38\theta^2 - 94\theta + 62$$



3.7 학습률이 크다면

$$J(\theta) = 38\theta^2 - 94\theta + 62$$



4 다변수 데이터에 대한 회귀

4.1 여러개의 특성(feature)

주택 규모 (x_1)	방 개수 (x_2)	화장실 개수 (x_3)	준공년도 (x_4)	주택 가격(y)
220	4	1	1994	325
135	3	2	2005	295
85	3	2	2012	250
55	2	1	2009	176
...



Multivariate Linear Regression 문제로 일반화할 수 있음



4.2 행렬식으로 표현

입력 변수(Feature)가 4개면?

$$h_{\theta}(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

$$= [\theta_0 \quad \theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4] \begin{bmatrix} x_0 = 1 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$h_{\theta}(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x}$$

5 간단한 예제 하나 - Boston 집값 예측

5.1 보스톤 집 가격 데이터

1 BOSTON 집값 데이터



✓ 이 데이터 세트는 Carnegie Mellon University에서 유지 관리함

→ The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81–102, 1978.

✓ 보스턴 주택 가격 데이터는 회귀 문제를 다루는 많은 머신 러닝 논문에서 사용함

5.2 데이터 읽기

```
from sklearn.datasets import load_boston

boston = load_boston()
print(boston.DESCR)

.. _boston_dataset:

Boston house prices dataset
-----
**Data Set Characteristics:**

:Number of Instances: 506
```

5.3 각 특성의 의미

```
1 [each for each in boston.feature_names]
```

```
['CRIM',
'ZN',
'INDUS',
'CHAS',
'NOX',
'RM',
'AGE',
'DIS',
'RAD',
'TAX',
'PTRATIO',
'B',
'LSTAT']
```

- CRIM : 범죄율
- INDUS : 비소매상업지역 면적 비율
- NOX : 일산화질소 농도
- RM : 주택당 방 수
- LSTAT : 인구 중 하위 계층 비율
- B : 인구 중 흑인 비율
- PTRATIO : 학생/교사 비율
- ZN : 25,000 평방피트를 초과 거주지역 비율
- CHAS : 찰스강의 경계에 위치한 경우는 1, 아니면 0
- AGE : 1940년 이전에 건축된 주택의 비율
- RAD : 방사형 고속도로까지의 거리
- DIS : 직업센터의 거리
- TAX : 재산세율

5.4 데이터 파악을 위해 pandas로 정리

```
import pandas as pd

boston_pd = pd.DataFrame(boston.data, columns=boston.feature_names)
boston_pd['PRICE'] = boston.target

boston_pd.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2

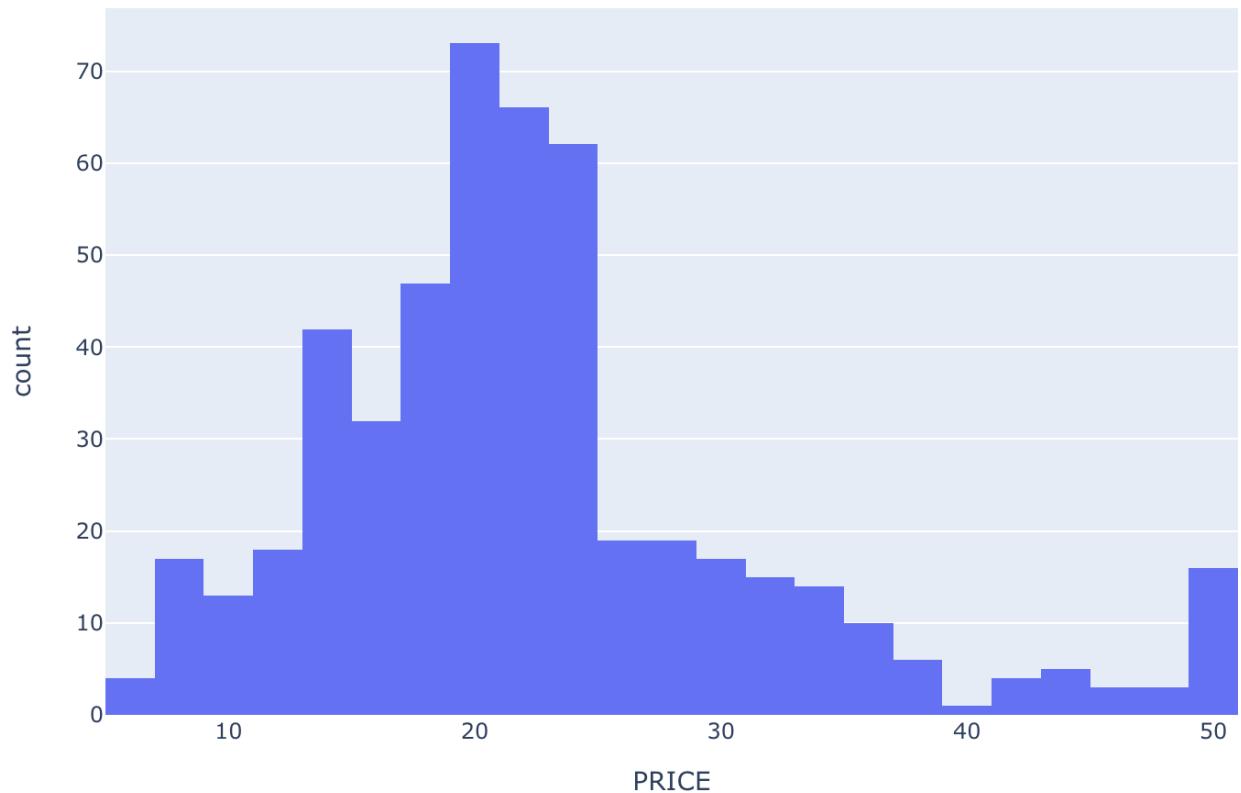
- 주의 - PRICE 컬럼은 Label이므로 이후 과정에서 잘 다루어야 한다

5.5 Price에 대한 histogram

```
import plotly.express as px

fig = px.histogram(boston_pd, x="PRICE")
fig.show()
```

5.6 집값에 대한 히스토그램

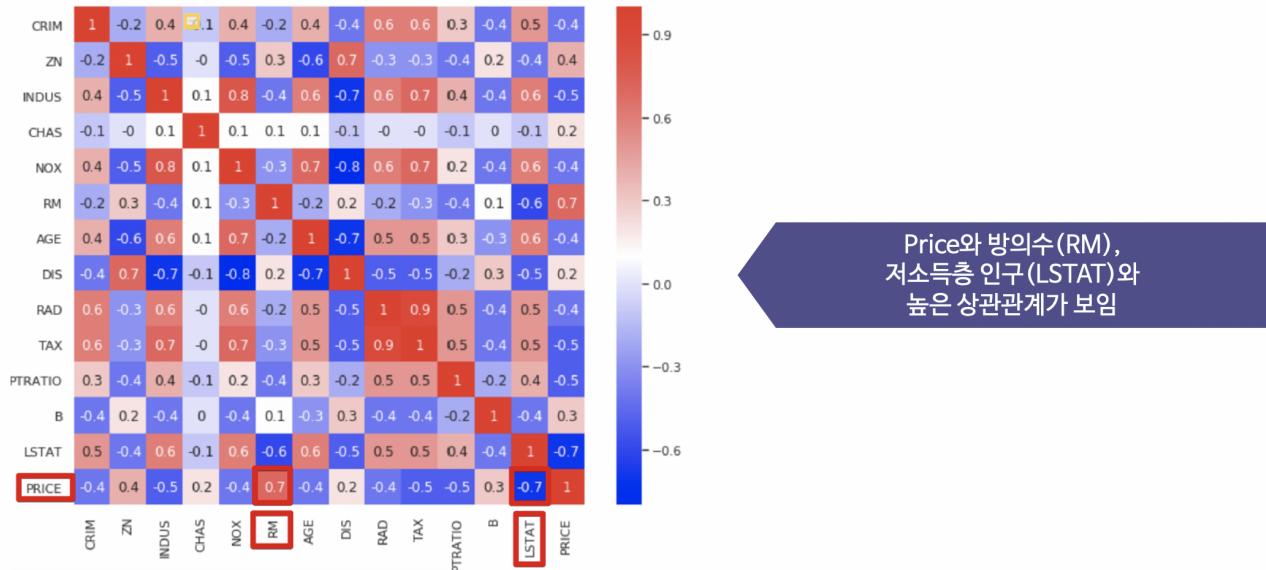


5.7 각 특성별 상관계수 확인

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

corr_mat = boston_pd.corr().round(1)
sns.set(rc={'figure.figsize':(10,8)})
sns.heatmap(data=corr_mat, annot=True, cmap='bwr');
```

5.8 상관관계 조사 결과



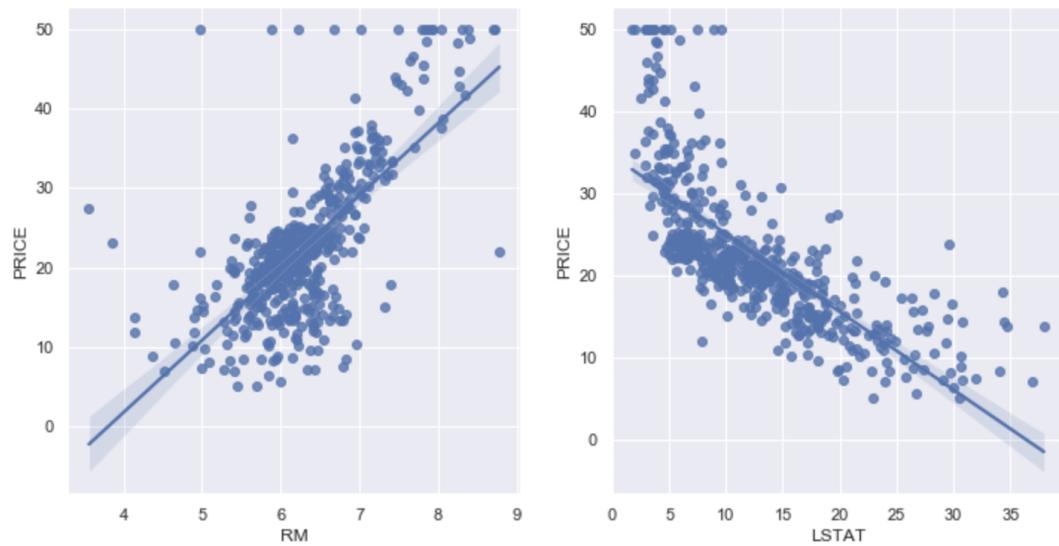
5.9 RM과 LSTAT와 PRICE의 관계에 대해 좀 더 관찰해보자

```

sns.set_style('darkgrid')
sns.set(rc={'figure.figsize':(12,6)})
fig, ax = plt.subplots(ncols=2)
sns.regplot(x='RM', y='PRICE', data=boston_pd, ax=ax[0])
sns.regplot(x='LSTAT', y='PRICE', data=boston_pd, ax=ax[1]);

```

5.10 저소득층 인구가 낮을 수록, 방의 갯수가 많을 수록 집 값이 높아짐???



- 이 가설은 문제가 없을까?

5.11 일단 데이터를 나누고

```
from sklearn.model_selection import train_test_split

X = boston_pd.drop('PRICE', axis=1)
y = boston_pd['PRICE']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=13)
```

5.12 LinearRegression을 사용하고

```
from sklearn.linear_model import LinearRegression

reg = LinearRegression()
reg.fit(X_train, y_train)

LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

5.13 모델 평가는 RMS로

```

import numpy as np
from sklearn.metrics import mean_squared_error

pred_tr = reg.predict(X_train)
pred_test = reg.predict(X_test)
rmse_tr = (np.sqrt(mean_squared_error(y_train, pred_tr)))
rmse_test = (np.sqrt(mean_squared_error(y_test, pred_test)))

print('RMSE of Train Data : ', rmse_tr)
print('RMSE of Test Data : ', rmse_test)

```

RMSE of Train Data : 4.642806069019824
RMSE of Test Data : 4.9313525841467

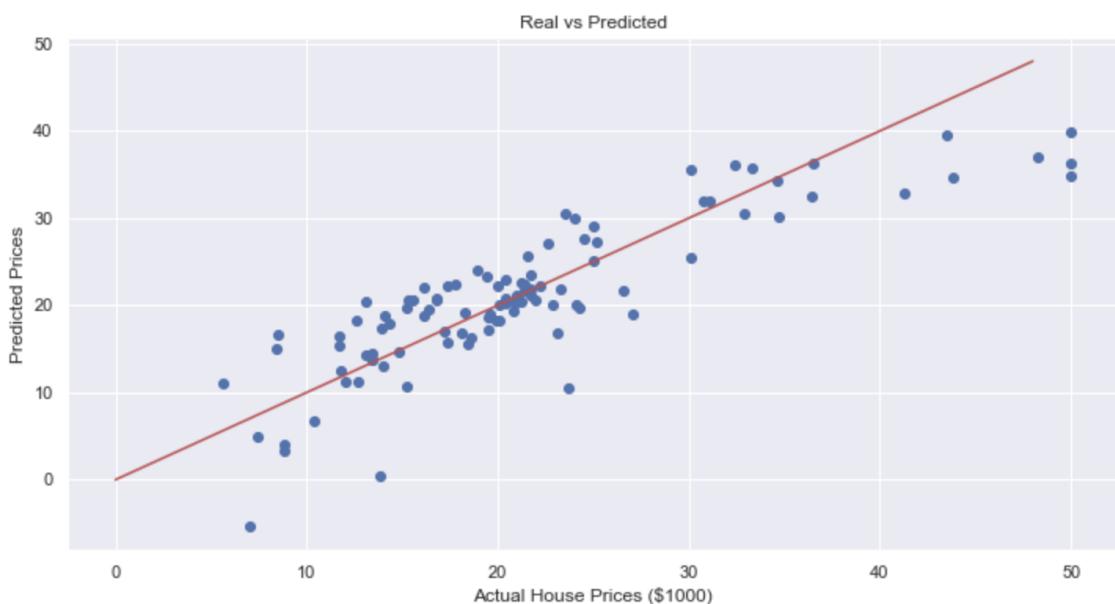
5.14 성능 확인

```

plt.scatter(y_test, pred_test)
plt.xlabel("Actual House Prices ($1000)")
plt.ylabel("Predicted Prices")
plt.title("Real vs Predicted")
plt.plot([0,48], [0,48], 'r')
plt.show()

```

5.15 결과



5.16 그런데, LSTAT를 사용하는 것이 맞는 걸까?

```
X = boston_pd.drop(['PRICE', 'LSTAT'], axis=1)
y = boston_pd['PRICE']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=13)

reg = LinearRegression()
reg.fit(X_train, y_train)

LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

5.17 당연히 성능은 나빠진다

```
pred_tr = reg.predict(X_train)
pred_test = reg.predict(X_test)
rmse_tr = (np.sqrt(mean_squared_error(y_train, pred_tr)))
rmse_test = (np.sqrt(mean_squared_error(y_test, pred_test)))

print('RMSE of Train Data : ', rmse_tr)
print('RMSE of Test Data : ', rmse_test)
```

RMSE of Train Data : 5.165137874244864
RMSE of Test Data : 5.295595032597165

5.18 이런 류의 고민은 아마 계속될 것~

```
plt.scatter(y_test, pred_test)
plt.xlabel("Actual House Prices ($1000)")
plt.ylabel("Predicted Prices")
plt.title("Real vs Predicted")
plt.plot([0,48], [0,48], 'r')
plt.show()
```

