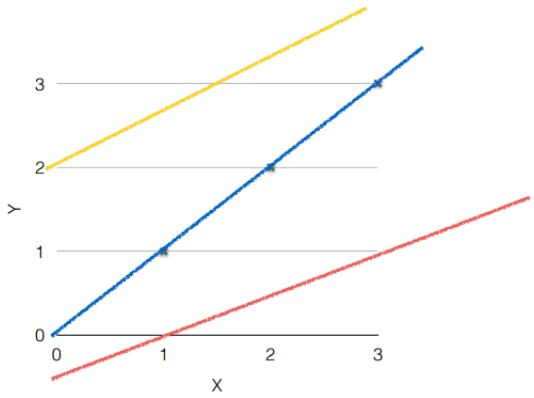
[선형회귀 (Linear regression)

단순 선형 회귀 - (단일변수)

(Linear) Hypothesis : H(x) = W x + b

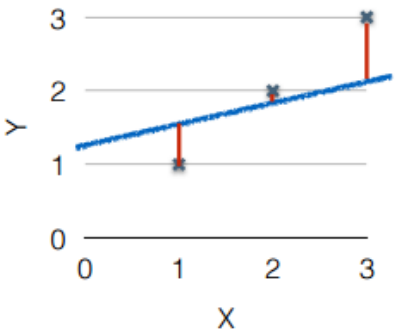
Which hypothesis is better?



minimize cost (W, b)

비용(Cost) 또는 손실(loss) : 예측 값 - 실제 값

$H(x) - y$



비용(Cost) 또는 손실(loss) 평균

$$\frac{(H(x^{(1)}) - y^{(1)})^2 + (H(x^{(2)}) - y^{(2)})^2 + (H(x^{(3)}) - y^{(3)})^2}{3}$$

$$cost = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$$

H(x) = W x + b 의 비용(Cost) 함수

$$cost(W, b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$$

1) 비용(Cost)/손실(loss) 함수

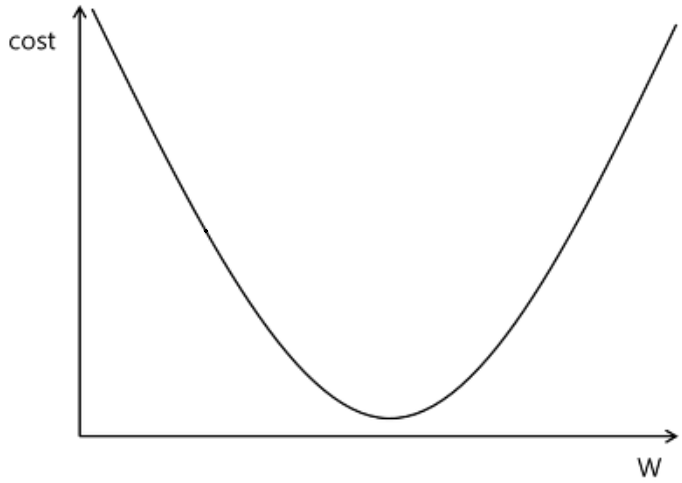
$$cost(W, b) = \frac{1}{m}\sum_{i=1}^{m}(H(x^{(i)}) - y^{(i)})^2$$

2) 비용 최소화를 위한 기법 : 경사하강법

머신 러닝의 목표 : minimize cost

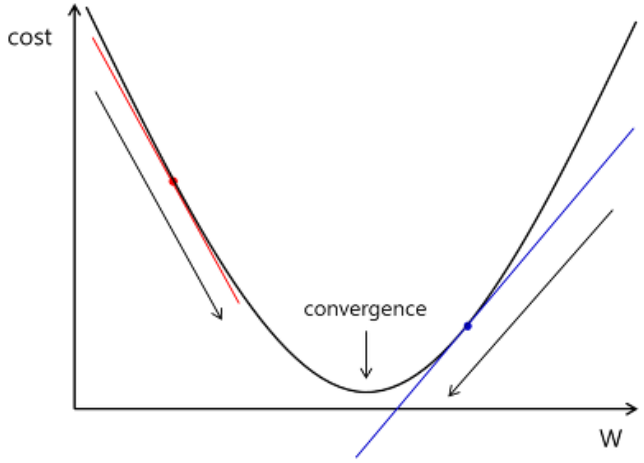
[경사하강법(Gradient descent algorithm) 이란?]

What cost(W) looks like?



cost를 줄이기 위해 변경되는 W의 파라미터의 상관관계를 그래프로 나타낸다면,
cost의 값이 최소가 되는 것은 W의 값이 가운데로 수렴하게 된다는 것.
(편의상 추가적으로 더하는 항인 바이어스의 값은 제외)

기울기가 0인 곳을 찾는 것이 경사하강법



Leeji

(핵심 함수)

```
model = LinearRegression()      # 학습 모델 선택
model = model.fit(x, y)         # 학습
result =model.predict([[7]])  # 예측
```

ex01_regression.py (단순 선형 회귀) x: [공부시간]

```
from sklearn.linear_model import LinearRegression

x = [[10],[5],[9],[7]]       #공부시간 10시간   5시간, 9시간, 7시간
y = [[100],[50],[90],[77]]  #시험점수 100점    50점,  90점   77점

model = LinearRegression()

model = model.fit(x, y)
result =model.predict([[7]])

print(result)
```

실행 결과:

[[72.01694915]]

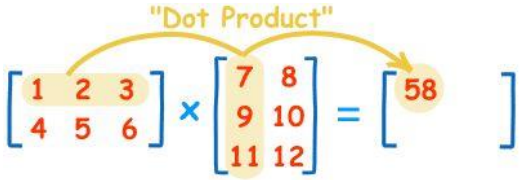
[(Multi-variable) linear regression]

다중선형 회귀 - (다중변수)

H(x1,x2,x3) = (x1w1) + (x2w2) + (x3w3)

$$\begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} x_1 w_1 + x_2 w_2 + x_3 w_3 \end{pmatrix} \qquad H(X) = XW$$

행렬곱 예)



국어 영어 수학

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} x_{11}w_1 + x_{12}w_2 + x_{13}w_3 \\ x_{21}w_1 + x_{22}w_2 + x_{23}w_3 \\ x_{31}w_1 + x_{32}w_2 + x_{33}w_3 \\ x_{41}w_1 + x_{42}w_2 + x_{43}w_3 \\ x_{51}w_1 + x_{52}w_2 + x_{53}w_3 \end{pmatrix}$$

[5, 3] [3, 1] [5, 1]

독립 변수 x와 이에 대응하는 종속 변수 y간의 관계가 다음과 같은 선형 함수 H(x)이면
선형 회귀분석(linear regression analysis)이라고 한다.

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_D x_D = w_0 + w^T x$$

위 식에서 w0,⋯,wD 를 함수 H(x)의 계수(coefficient)이자 이 선형 회귀모형의 parameter 라고 한다.

Leeji

ex02_regression.py (다중 선형 회귀) x: [공부시간, 학년]

```
from sklearn.linear_model import LinearRegression

x = [[10,3],[5,2],[9,3],[7,3]]      #공부시간,학년: 10시간,3  5시간,2, 9시간,3, 7시간,3
y = [[100],[50],[90],[77]]          #시험점수:      100점      50점,    90점      77점

model = LinearRegression()

model = model.fit(x, y)
result =model.predict([[7,2]])   #7시간공부,  2학년

print(result)
```

실행 결과:

[[65.]]

(데이터에 대한 사전 조사)

데이터에 결측치 또는 이상한 값(outlier) 이 있는지 확인
각 데이터가 연속적인 실수값인지 범주형 값인지 확인
실수형 데이터의 분포가 정규 분포인지 확인
실수형 데이터에 양수 혹은 범위 등으로 제한 조건이 있는지 확인
범주형 데이터의 경우 범주의 값이 어떤 값 혹은 숫자로 표현되어 있는지 확인
데이터 간의 상관관계를 확인

sklearn에 예제 데이터 중 보스턴 집값 예측 예

scikit-learn 이 제공하는 회귀 분석용 예제 데이터 중 하나인 보스턴 주택 가격 데이터에 대해 소개한다. 이 데이터는 다음과 같이 구성되어 있다.

- 타겟 데이터
    - 1978 보스턴 주택 가격
    - 506 타운의 주택 가격 중앙값 (단위 1,000 달러)

- 특징 데이터
    - 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: 1940 년 이전에 건축된 주택의 비율
    - RAD: 방사형 고속도로까지의 거리
    - DIS: 직업센터의 거리
    - TAX: 재산세율

ex03_regression_boston.py

```
from sklearn.datasets import load_boston
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

boston = load_boston()
dfX = pd.DataFrame(boston.data, columns=boston.feature_names)
dfy = pd.DataFrame(boston.target, columns=["MEDV"])
df = pd.concat([dfX, dfy], axis=1)
print( df.head())
```

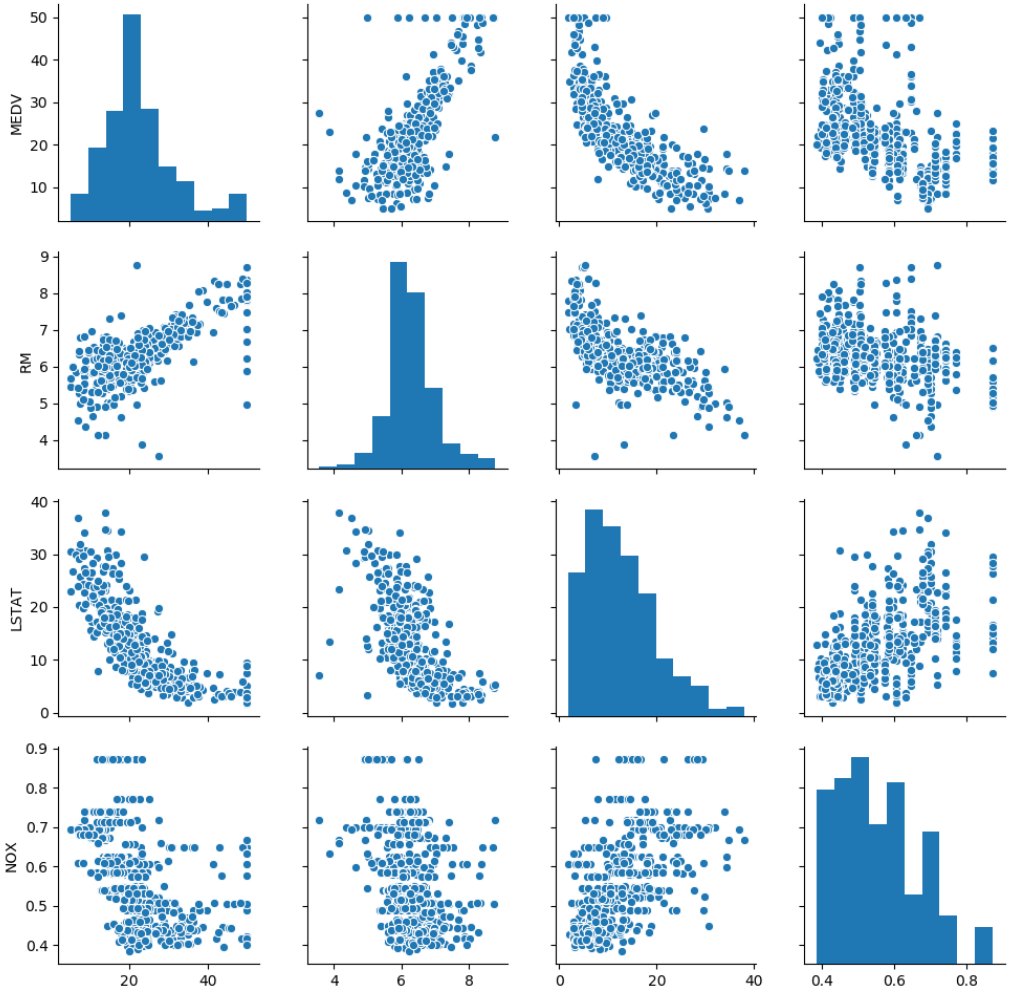
```
cols = ["MEDV", "RM", "LSTAT", "NOX"]
sns.pairplot(df[cols])
plt.show()
"""
가격(MEDV)과 RM 데이터가 강한 양의 상관관계,
LSTAT, NOX 데이터와 강한 음의 상관관계
"""
data = df[[ "RM","LSTAT","NOX"]]
label = df["MEDV"]

model = LinearRegression( )
model = model.fit(data, label)

predict = model.predict([[6, 9.67, 0.573 ]]) # RM(방 수):6개,   LSTAT:9.67   NOX: 0.573
print("예측 집값 : ", predict)
```

실행결과 :

	CRIM	ZN	INDUS	CHAS	NOX	...	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	...	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	...	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	...	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	...	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	...	222.0	18.7	396.90	5.33	36.2



예측 집값 : [22.89854372]