



---

# 파이썬을 이용한 머신러닝

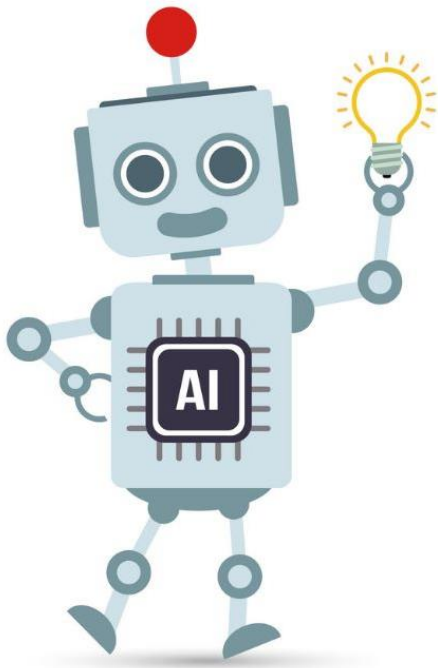
---

2019 / 06 / 25

정 윤 주

# 목 차

• SW중심대학



## ○ 소프트웨어 중심대학 교육 설계

- 전공자 SW 교육 설계
- 비전공자 SW 교육 설계
- 지역민들을 위한 SW 교육 설계

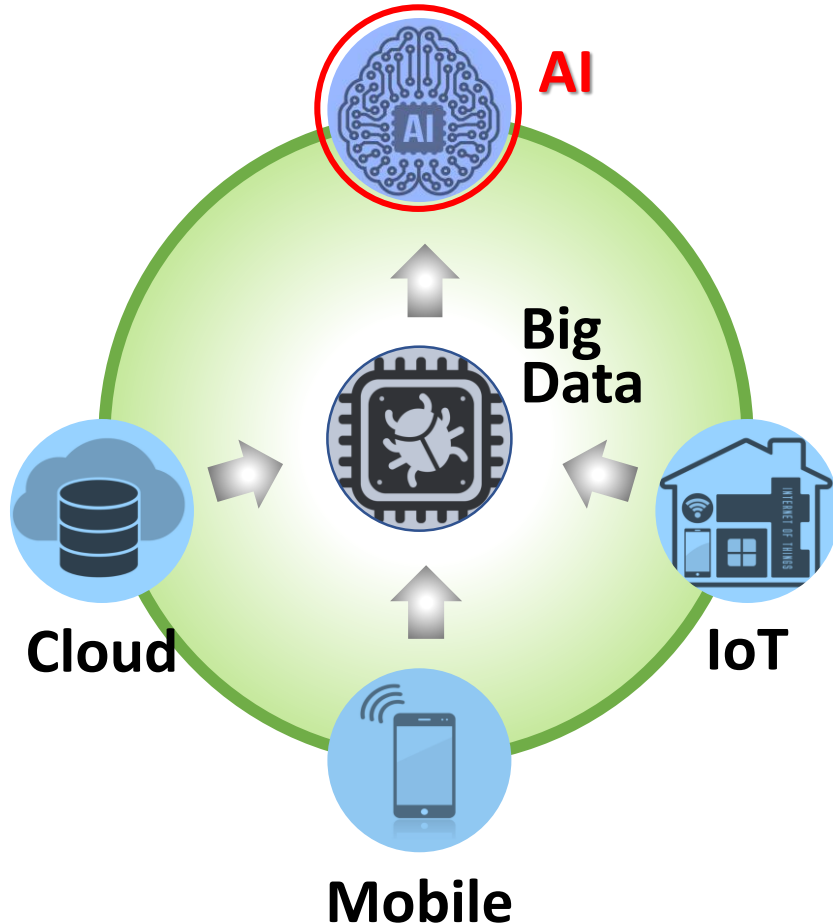
## ○ 파이썬을 이용한 머신러닝

- 선형 회귀와 학습
- 비용과 경사 하강
- 선형 회귀 인공지능(AI) 구현
- 요약

# 전공자 SW교육 설계

• SW중심대학

4차 산업혁명 이후 글로벌 소프트웨어 개발 핵심 인재의 양성



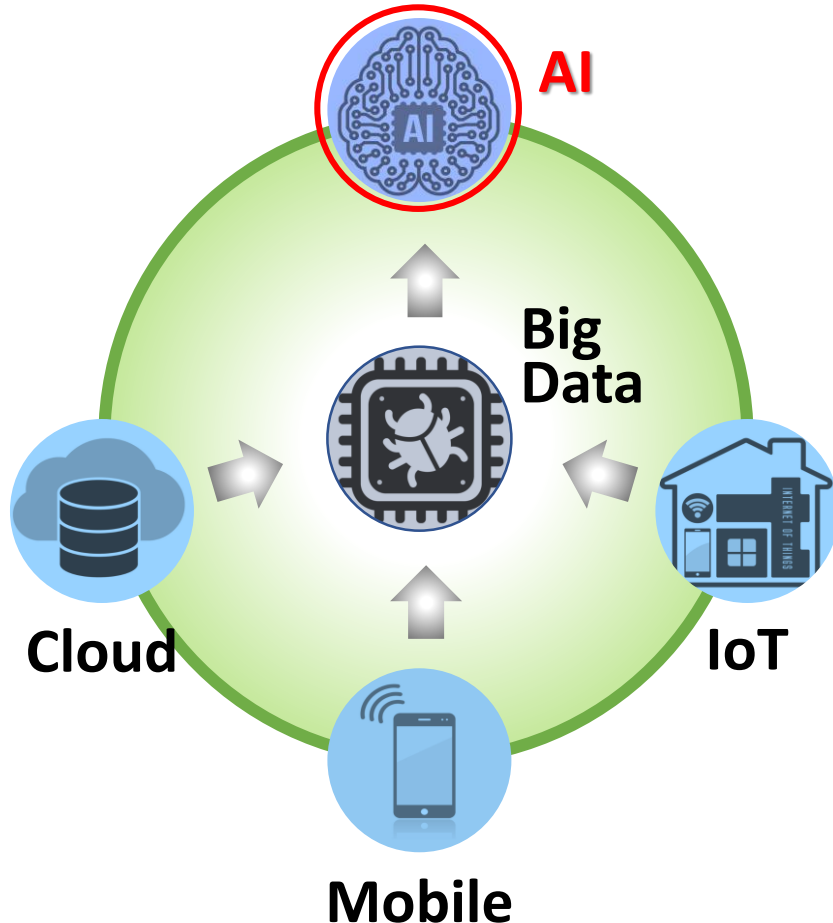
AI	IoT	Data Science	Mobile	Cloud
Python				
Java / Javascript				
C / C++				
R		R		클라우드 컴퓨팅언어  • SaaS PaaS IaaS
	라즈베리파이	SQL	앱인벤터	
	아두이노	SAS	Objective-C	
			Swift	

Open Source System(OSS)를 통한 협업학습 및  
분산 버전 관리 → **우수한 SW개발자로 양성**

# 전공자 SW교육 설계

• SW중심대학

4차 산업혁명 이후 글로벌 소프트웨어 개발 핵심 인재의 양성



AI	IoT	Data Science	Mobile	Cloud
Python				
Java / Javascript				
C / C++				
R		R		클라우드 컴퓨팅언어 • SaaS PaaS IaaS
	라즈베리파이	SQL	앱인벤터	
	아두이노	SAS	Objective-C	
			Swift	

Open Source System(OSS)를 통한 협업학습 및  
분산 버전 관리 → **우수한 SW개발자로 양성**





























- SW중심대학

: 창의적 사고와 코딩을 통하여 "창의적·실제적" 문제 해결 역량 강화를 목표



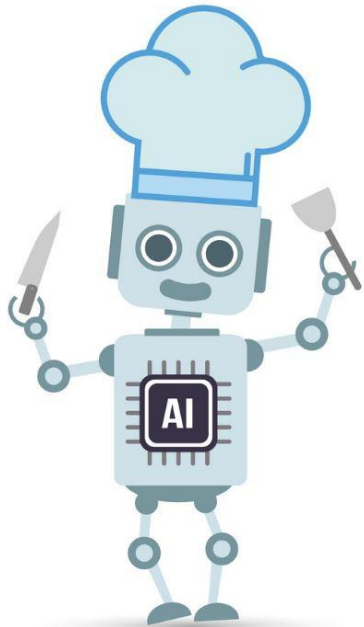
# 지역민들을 위한 SW 교육 설계

SW중심대학

	아동·청소년	경력단절 여성	교사 및 교육종사자	기업인·농업인
초			  	  
중	 	 	 	 
고	  	   	 	  

# 파이썬을 이용한 머신러닝

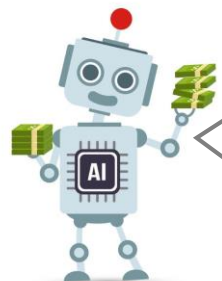
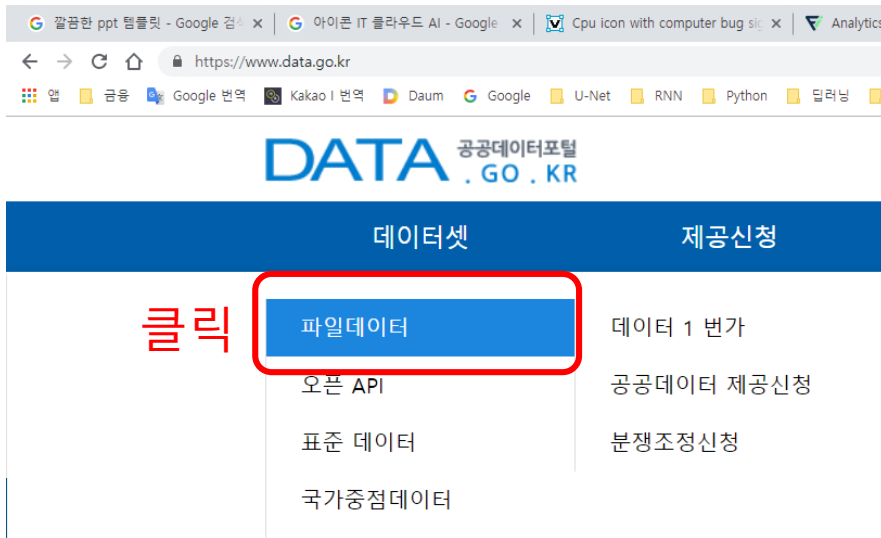
## 학습 목표



- ☐ 선형 회귀를 이해할 수 있다.
- ☐ 머신러닝의 학습(Learning)을 이해할 수 있다.
- ☐ 비용함수를 이해할 수 있다.
- ☐ 최적의 학습 결과를 찾는 과정을 이해할 수 있다.
- ☐ 텐서플로우를 활용하여 머신러닝을 구현할 수 있다.

# (복습)파이썬을 이용한 데이터 분석

## 1. 공공 데이터 가져오기 <https://www.data.go.kr/>



복습!!

- 공공 데이터 가져오기
- 그래프 그리기



# (복습)파이썬을 이용한 데이터 분석

## 1. 공공 데이터 가져오기

☐ CSV 고속도로\_상습정체구간(2015년11월...



멀티다운로드



상세정보



오류신고



☐

CSV

고속도로\_노선별 월 변동계수(2015년...



멀티다운로드



상세정보



오류신고



☐ CSV 고속도로\_노선별 요일 변동계수(2015...



다운로드



상세정보



오류신고



☐

CSV

고속도로\_지정차로제지정현황(2015년)



다운로드



상세정보



오류신고



☐ CSV 고속도로\_교통사고통계(2015년11월...



다운로드



상세정보



오류신고



☐

CSV

고속도로 시공간 분산지수(2016년)



다운로드



상세정보



오류신고



→ “고속도로교통사고현황.csv” 라는 이름으로 저장

# (복습)파이썬을 이용한 데이터 분석

## 2. 데이터 파일 읽기

```
import pandas as pd
import matplotlib.pyplot as plt

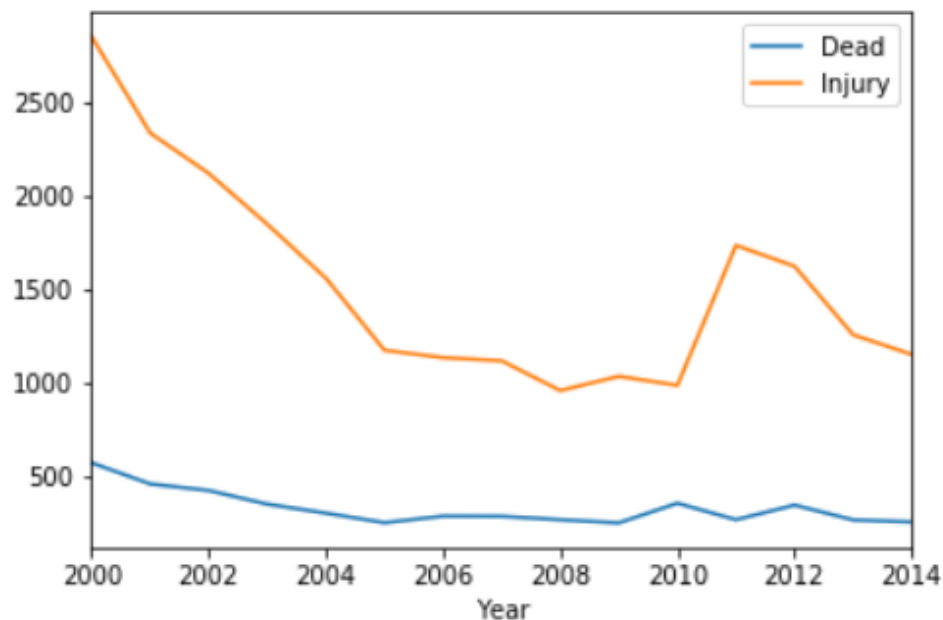
#df = pd.read_csv('고속도로교통사고현황.csv')
df = pd.read_csv('고속도로교통사고현황.csv', encoding='CP949')
df
```

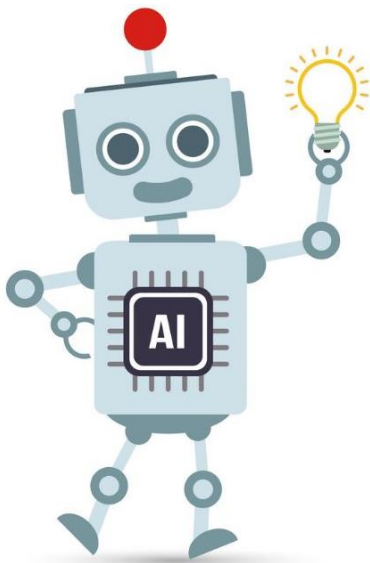
	연도	사고	사망	부상
0	2000	3910	569	2845
1	2001	3638	456	2331
2	2002	3957	421	2115
3	2003	3585	348	1843
4	2004	3242	300	1555
5	2005	2880	249	1170
6	2006	2583	284	1131
7	2007	2550	283	1114
8	2008	2449	265	955
9	2009	2374	248	1031
10	2010	2368	353	983
11	2011	2640	265	1731
12	2012	2600	343	1619
13	2013	2496	264	1253
14	2014	2395	253	1148

# (복습)파이썬을 이용한 데이터 분석

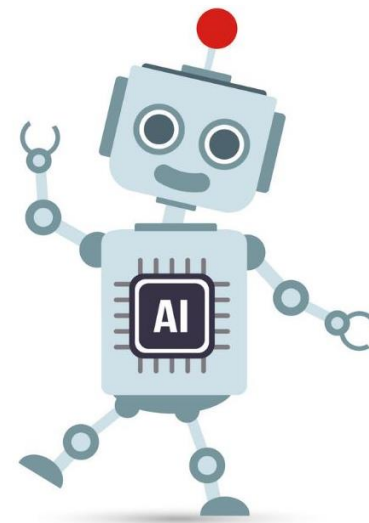
## 3. 데이터의 일부를 이용하여 시각화하기

```
df1=df.loc[:,['연도','사망','부상']]
ax=df1.plot(kind='line',x='연도')
ax.set_xlabel('Year')
ax.legend(['Dead','Injury'])
plt.show()
```





# 파이썬을 이용한 머신러닝



1. 선형 회귀(Linear Regression)와 학습(Learning)
2. 비용 (Cost)과 경사 하강(Gradient Descent)
3. 간단한 선형 회귀 인공지능 구현
4. 단원 요약

수업 자료는 <https://github.com/Yunju-Jeong/software> 에서 다운로드 받으세요.

# (파이썬을 이용한 머신러닝) 선형 회귀와 학습

• SW중심대학

## 기존 프로그램과 머신러닝의 차이

기존의 프로그램

데이터  
입력



처리 과정



데이터  
출력

# (파이썬을 이용한 머신러닝) 선형 회귀와 학습

• SW중심대학

## 기존 프로그램과 머신러닝의 차이

머신러닝

데이터  
입력



처리 과정



데이터  
출력

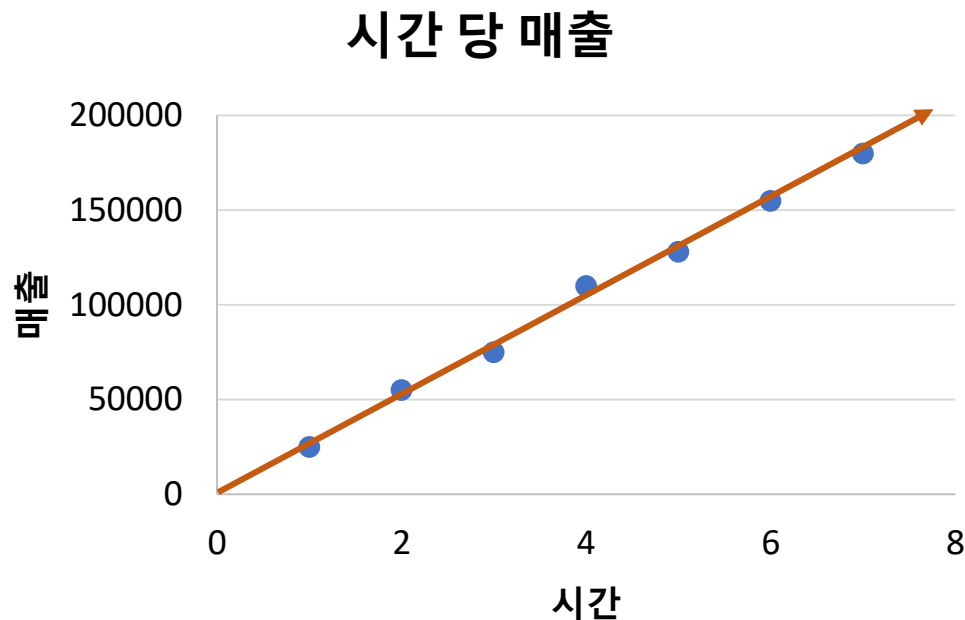
# (파이썬을 이용한 머신러닝) 선형 회귀와 학습

## 선형 회귀(Linear Regression)

: 변수 사이의 선형적인 관계를 모델링한 것

(예) 자영업자의 노동 시간과 하루 매출 데이터가 아래와 같다고 가정하자.

하루 노동 시간	하루 매출
1	25,000
2	55,000
3	75,000
4	110,000
5	128,000
6	155,000
7	180,000



# (파이썬을 이용한 머신러닝) 선형 회귀와 학습

• SW중심대학

## 선형 회귀와 학습

**일상 생활의 많은 현상들은 선형적인 성격을 가진다.**

- 선형 회귀 : 선형적인 관계에 적용하는 대표적인 기계학습이론

**학습을 시킨다 = 선형 회귀 모델을 구축한다**

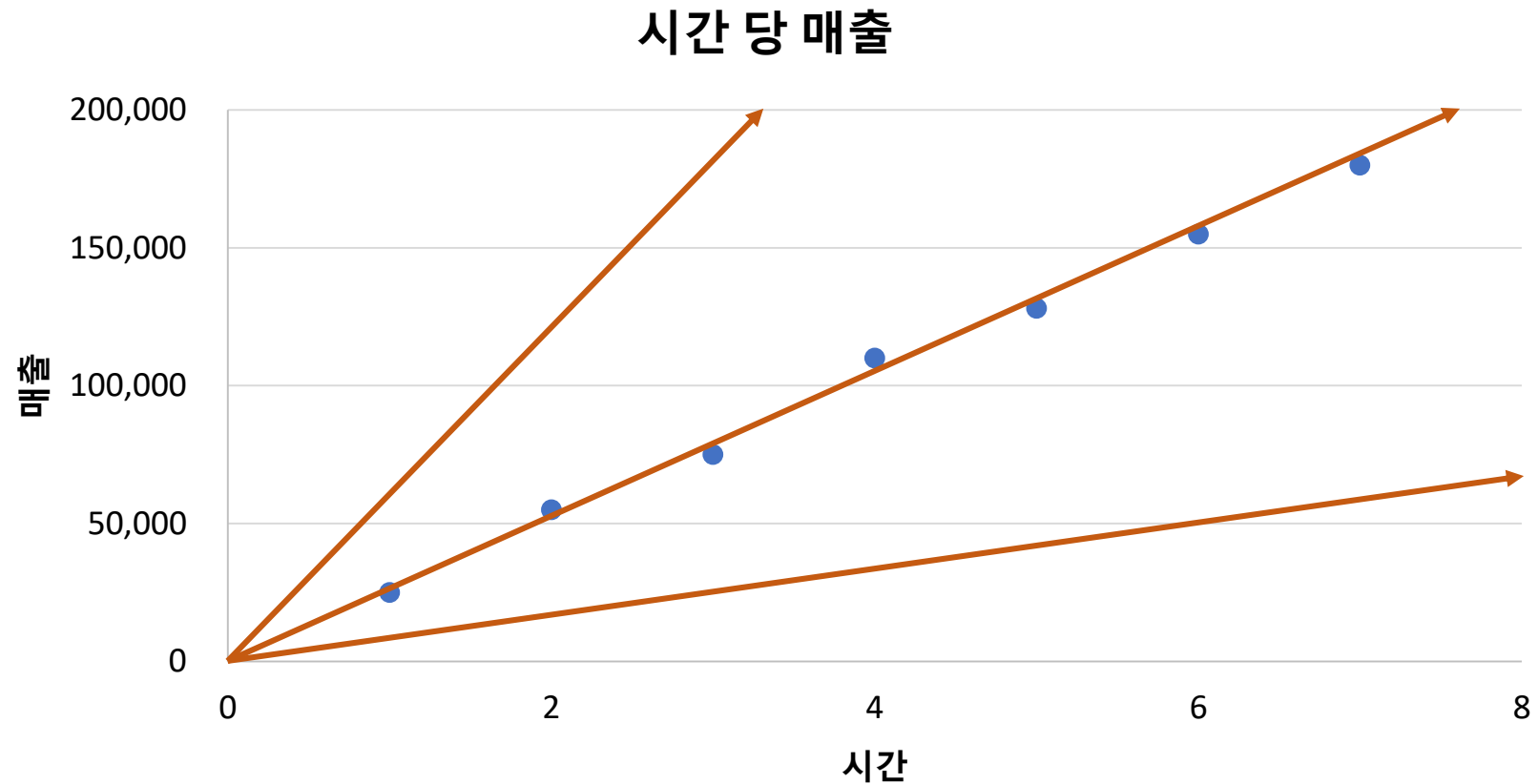
- 주어진 데이터를 학습시켜서 가장 합리적인 '직선'을 찾아낸다.
- 데이터는 3개 이상일 때 의미가 있다.



# (파이썬을 이용한 머신러닝) 선형 회귀와 학습

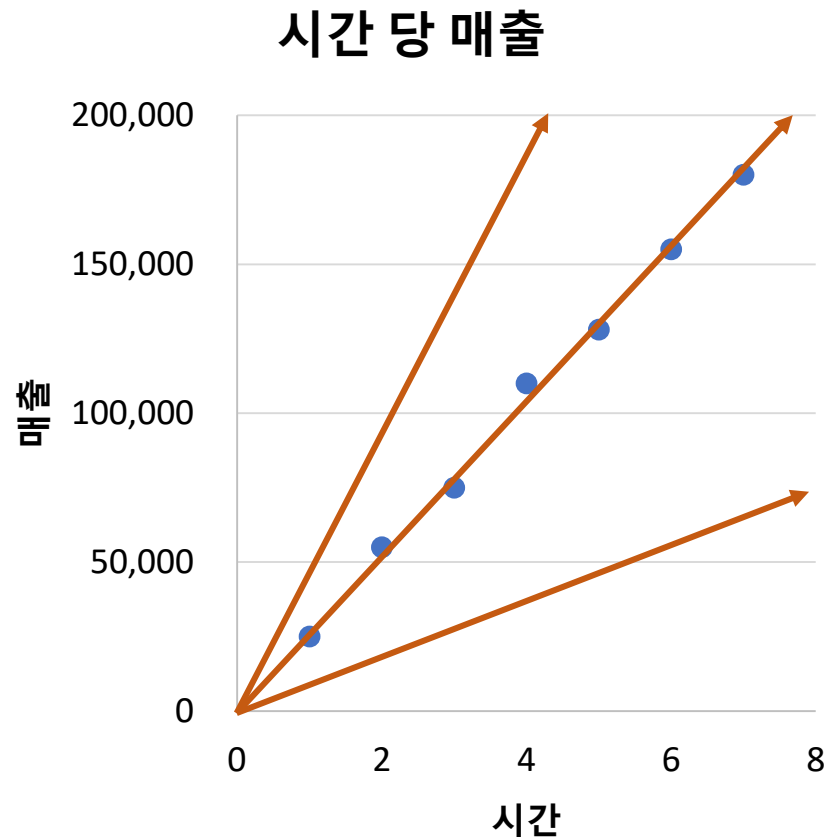
• SW중심대학

## 가장 합리적인 선은?



# (파이썬을 이용한 머신러닝) 선형 회귀와 학습

## 가장 합리적인 선은?



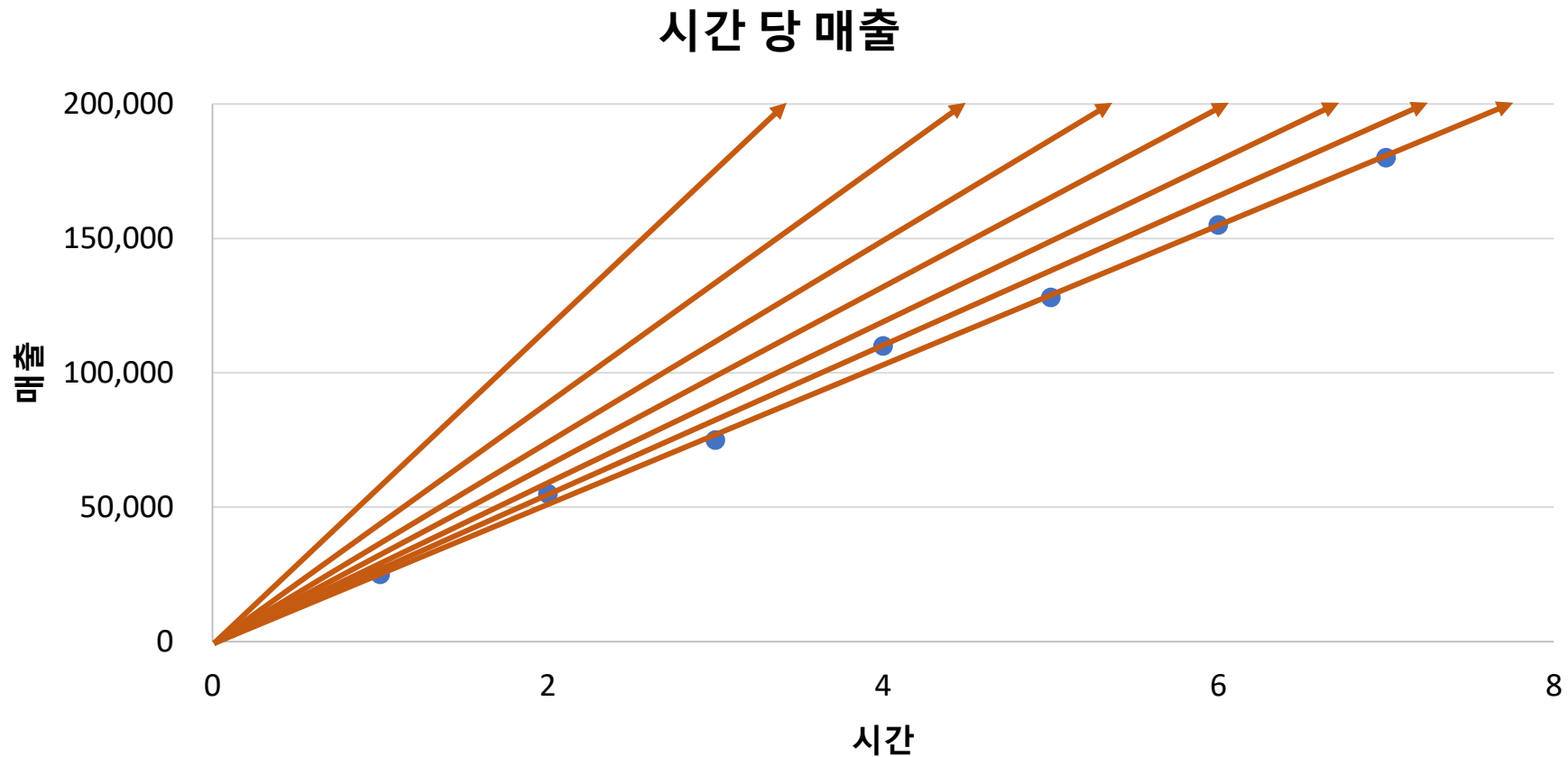
$$H(x) = W * x + b$$

- 일차 방정식을 이용하여 직선을 표현
  - 가설을 수정해 나가면서 가장 합리적인 식을 탐색
  - 선형 회귀란 주어진 데이터를 이용하여 일차 방정식을 수정해 나가는 것
    - 학습을 거쳐서 가장 합리적인 선을 찾아내는 것
    - 학습을 많이 해도 '완벽한' 식을 찾지 못할 수도 있다.
- 실제 사례에서는 근사값을 찾는 것 만으로도 충분할 때가 있다.

# (파이썬을 이용한 머신러닝) 선형 회귀와 학습

• SW중심대학

## 선형 회귀와 학습의 직관적인 이해

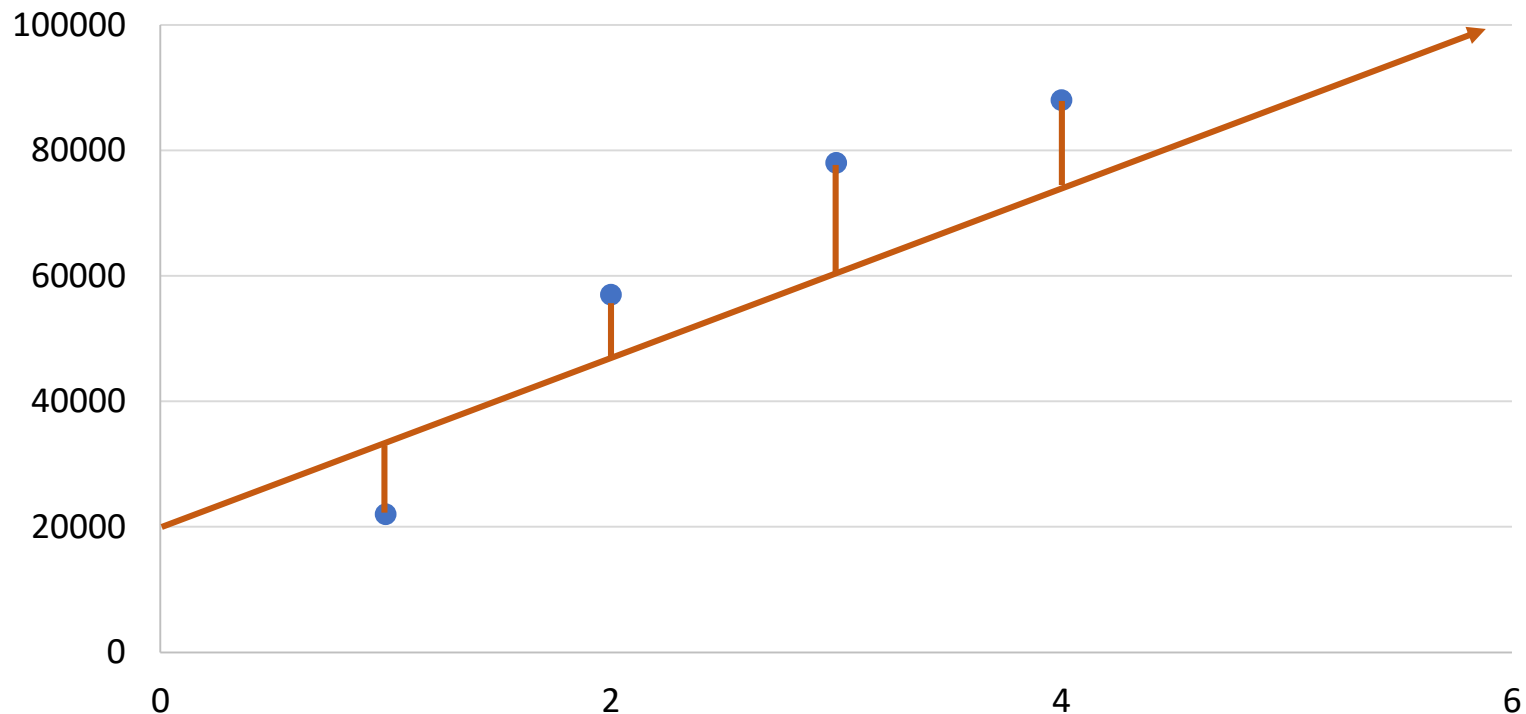


# (파이썬을 이용한 머신러닝) 비용과 경사하강

SW중심대학

## 비용(Cost)

: 가설이 얼마나 정확한 지 판단하는 기준



# (파이썬을 이용한 머신러닝) 비용과 경사하강

• SW중심대학

## 비용 함수(Cost Function)

### (예측 값 - 실제 값)<sup>2</sup>의 평균

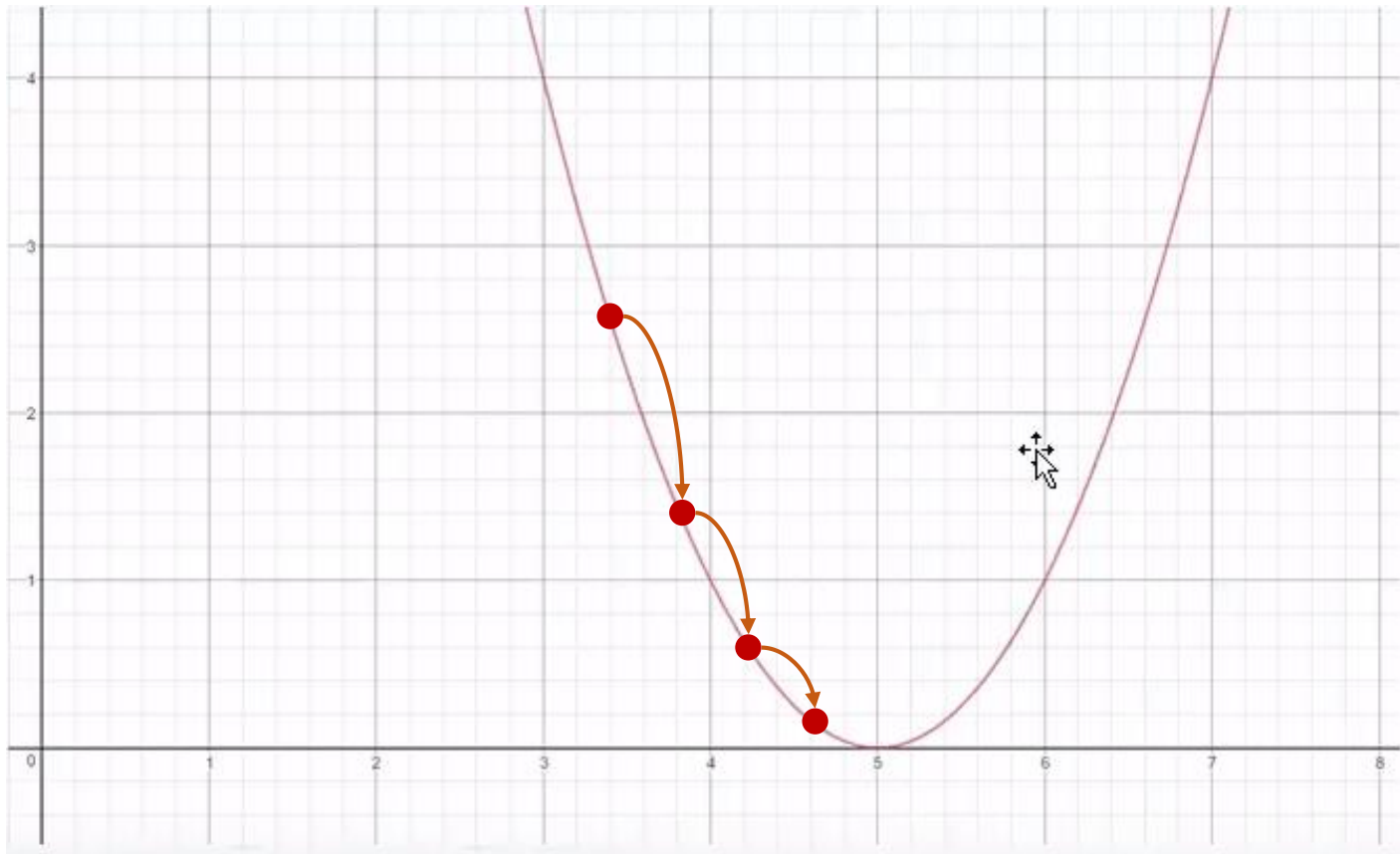
- 현재의  $w, b$  값과 데이터를 이용하면 비용함수를 구할 수 있다.
- 비용함수로 구한 비용이 적을 수록 좋다.

$$Cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x_i) - y_i)^2$$

# (파이썬을 이용한 머신러닝) 비용과 경사하강

• SW중심대학

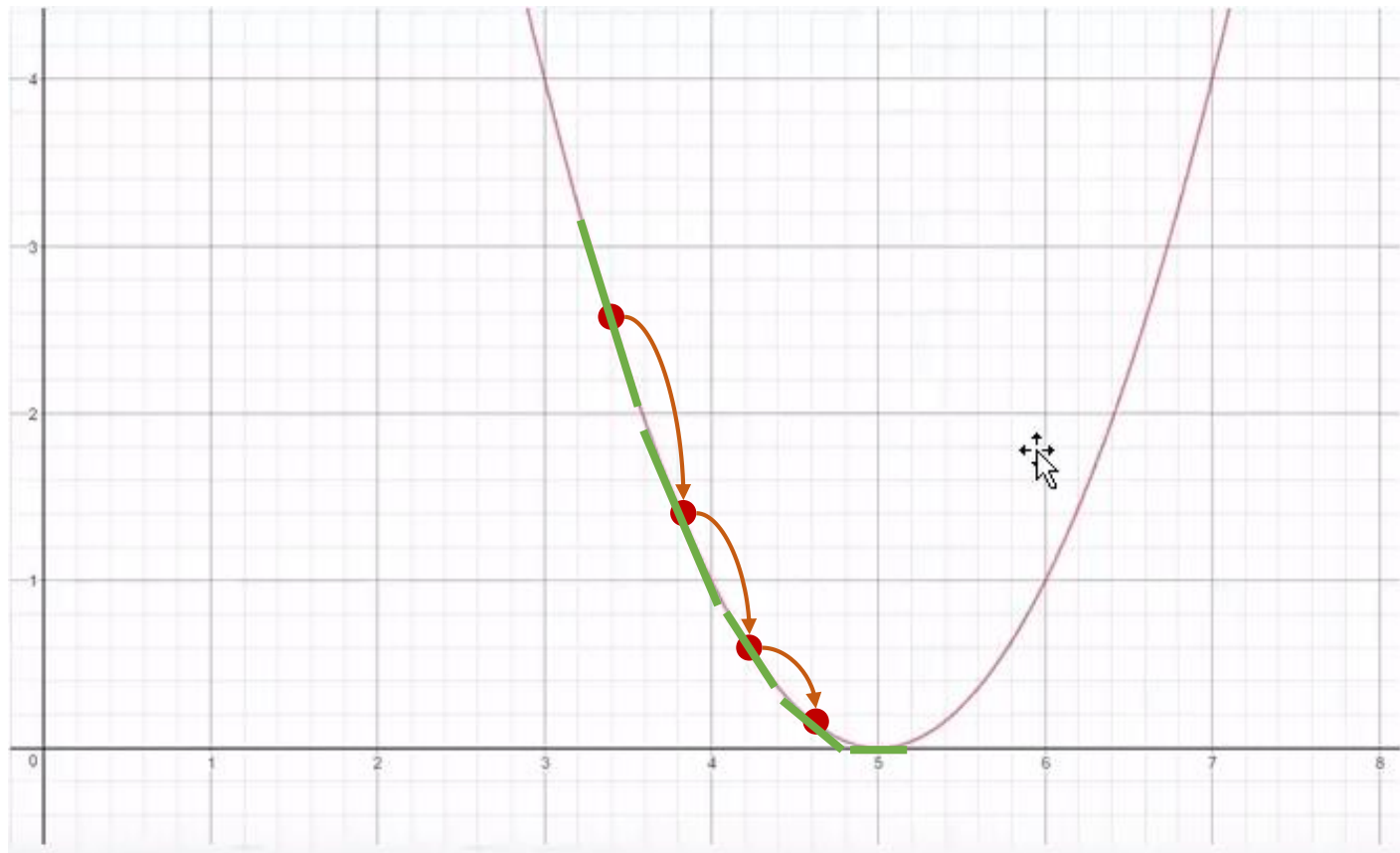
## 경사 하강(Gradient Descent)



# (파이썬을 이용한 머신러닝) 비용과 경사하강

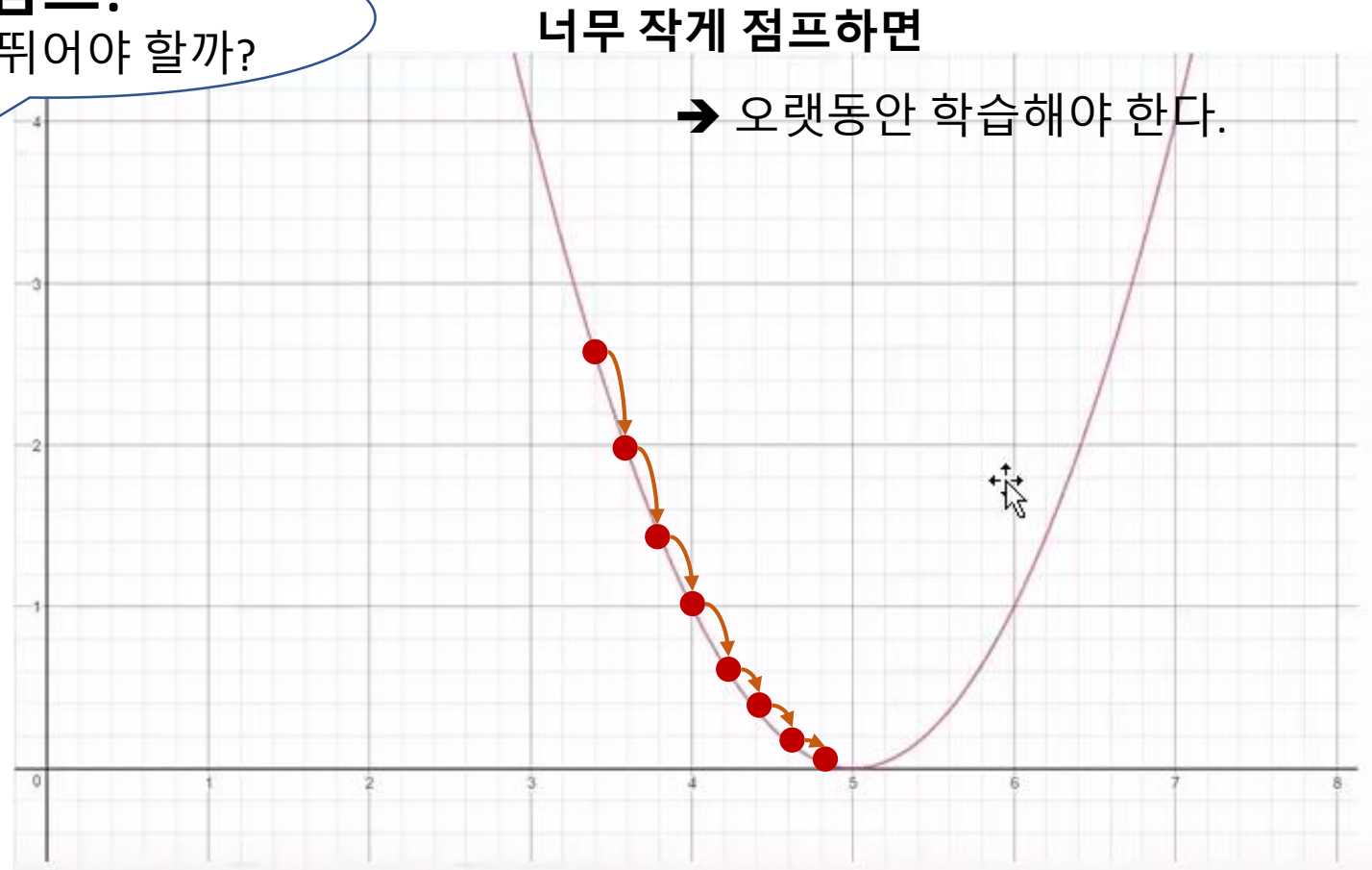
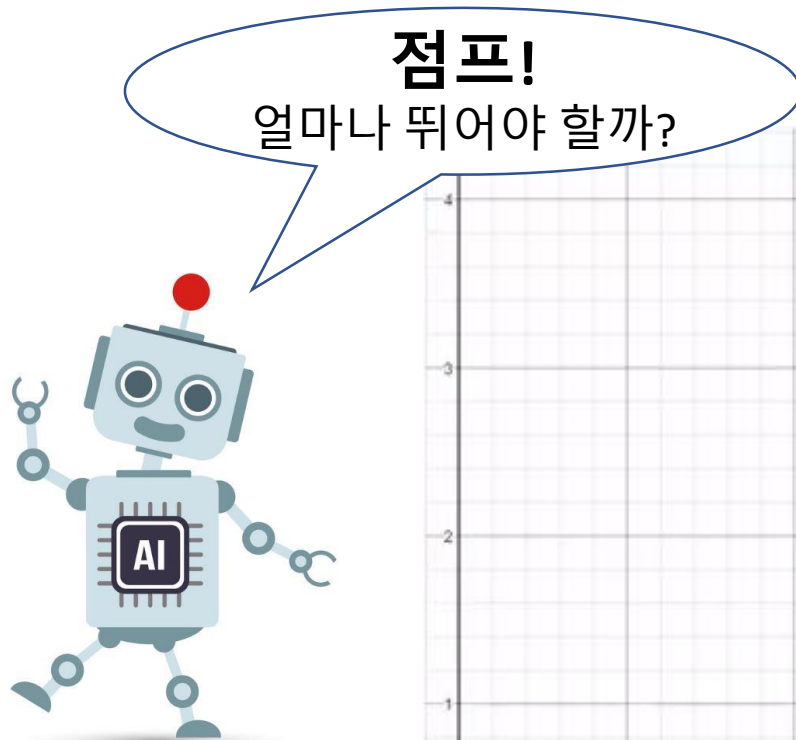
• SW중심대학

## 미분과 기울기



# (파이썬을 이용한 머신러닝) 비용과 경사하강

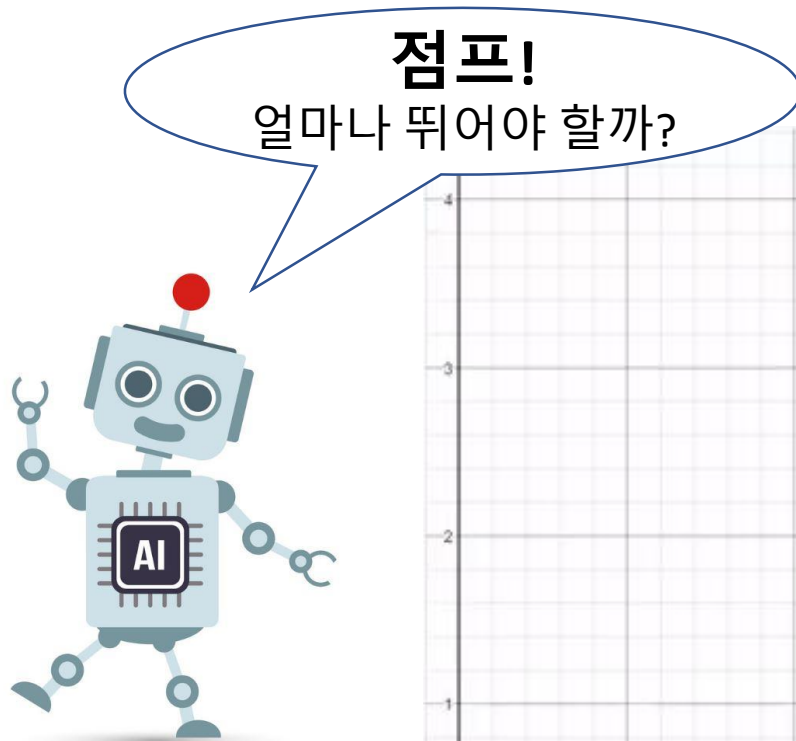
• SW중심대학





# (파이썬을 이용한 머신러닝) 비용과 경사하강

SW중심대학

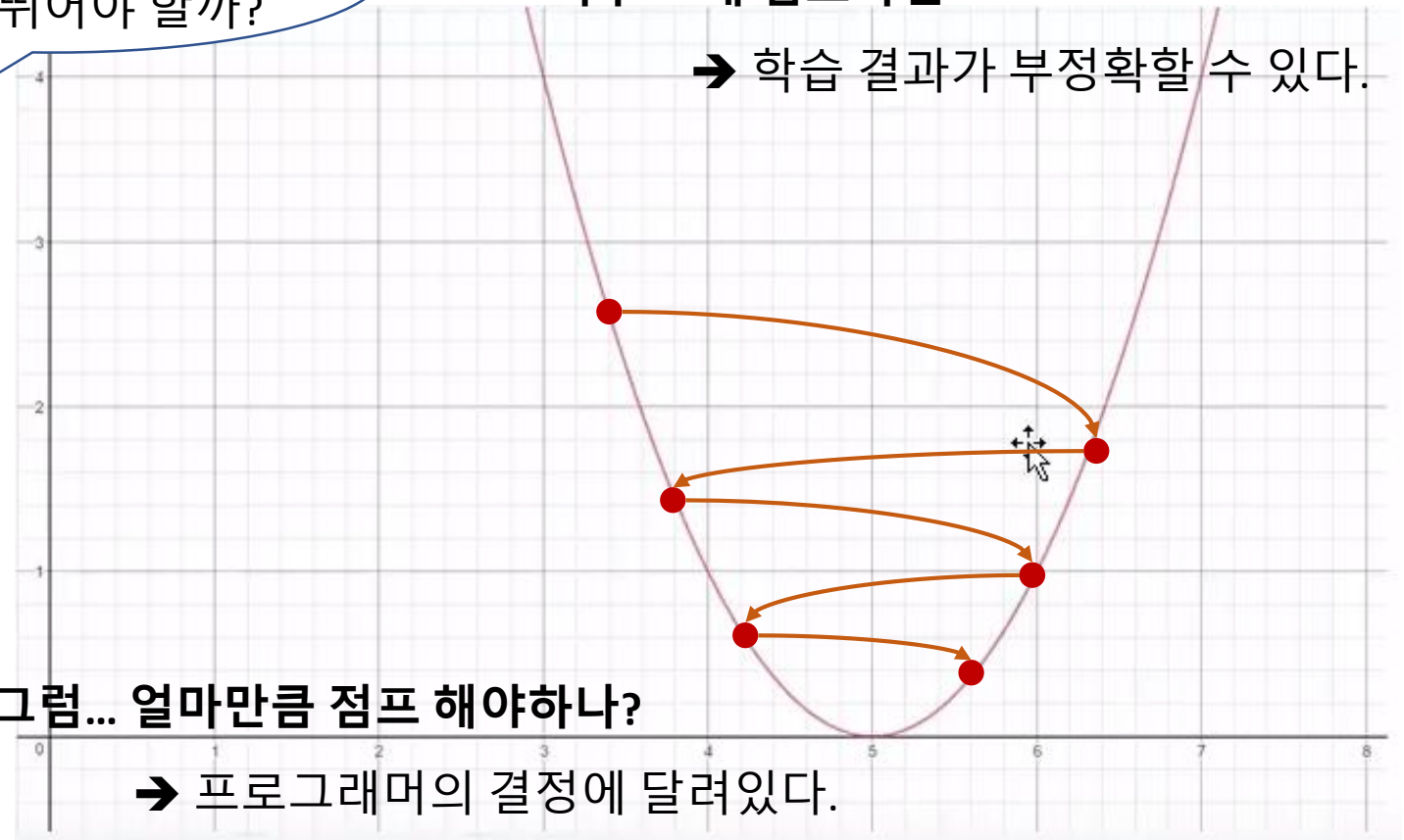


너무 크게 점프하면

→ 학습 결과가 부정확할 수 있다.

그럼... 얼마만큼 점프 해야하나?

→ 프로그래머의 결정에 달려있다.



# (파이썬을 이용한 머신러닝) 선형회귀 AI 구현

## 1. 머신러닝을 이용한 교통 사고 수 예측

```
*ML_usingPublicData.py - D:/jyj/2019강의관련/소프트웨어_강의전담/공개강의/ML_usingPublicData.py (3.5.4rc1)*
File Edit Format Run Options Window Help

import tensorflow as tf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('고속도로교통사고현황.csv', encoding='CP949')
ar = df.to_numpy()
xData = ar[:, 0]-2000 # 연도에서 2000을 뺀다. (예) 2010 ==> 10
yData = ar[:, 1]      # 사고 수

W = tf.Variable(tf.random_uniform([1], -100, 100)) # W 변수 선언, 초기값을 랜덤 넘버로 설정
b = tf.Variable(tf.random_uniform([1], -100, 100)) # b 변수 선언, 초기값을 랜덤 넘버로 설정
X = tf.placeholder(tf.float32)                   # X 변수 선언
Y = tf.placeholder(tf.float32)                   # Y 변수 선언
H = W * X + b                                     # 선형 회귀 방정식 선언, 가설 정
cost = tf.reduce_mean(tf.square(H-Y))           # 비용함수 정의
a = tf.Variable(0.01)                             # Learning rate, 경사하강 점프 스텝 초기화
optimizer = tf.train.GradientDescentOptimizer(a)  # 경사하강 라이브러리를 이용하여 최적화
train = optimizer.minimize(cost)                 # 비용이 최소화하는 방향으로 학습
```

# (파이썬을 이용한 머신러닝) 선형회귀 AI 구현

• SW중심대학

## 1. 머신러닝을 이용한 교통 사고 수 예측

```
init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init)
for i in range(5001):
    sess.run(train, feed_dict={X : xData, Y : yData}) # 실제로 학습을 진행
    if i % 500 == 0:
        print(i, sess.run(cost, feed_dict={X : xData, Y : yData}), sess.run(W), sess.run(b))

print(sess.run(H, feed_dict={X : [16]}))

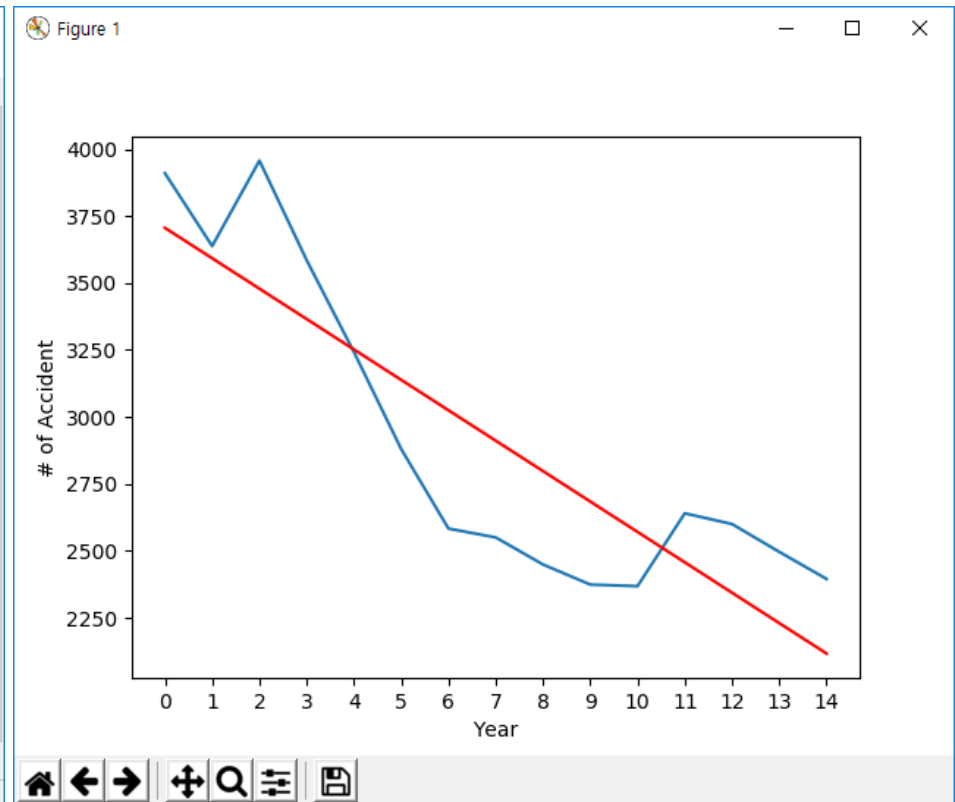
hY = sess.run([(W * x + b) for x in xData])
plt.plot(xData, yData)
plt.plot(xData, hY, 'red')
plt.show()
```

Ln: 6 Col: 10

# (파이썬을 이용한 머신러닝) 선형회귀 AI 구현

## 2. 실험 결과와 시각화

```
*Python 3.6.5 Shell*
File Edit Shell Debug Options Window Help
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/JYJ/ML_test01
/test.py =====
0 4239504.5 [340.8426] [-26.399437]
500 97475.76 [-88.37711] [3463.6643]
1000 81358.12 [-111.90616] [3690.1946]
1500 81290.484 [-113.4302] [3704.8674]
2000 81290.234 [-113.52895] [3705.818]
2500 81290.22 [-113.53346] [3705.8616]
3000 81290.22 [-113.53346] [3705.8616]
3500 81290.22 [-113.53346] [3705.8616]
4000 81290.22 [-113.53346] [3705.8616]
4500 81290.22 [-113.53346] [3705.8616]
5000 81290.22 [-113.53346] [3705.8616]
[1889.3262]
```

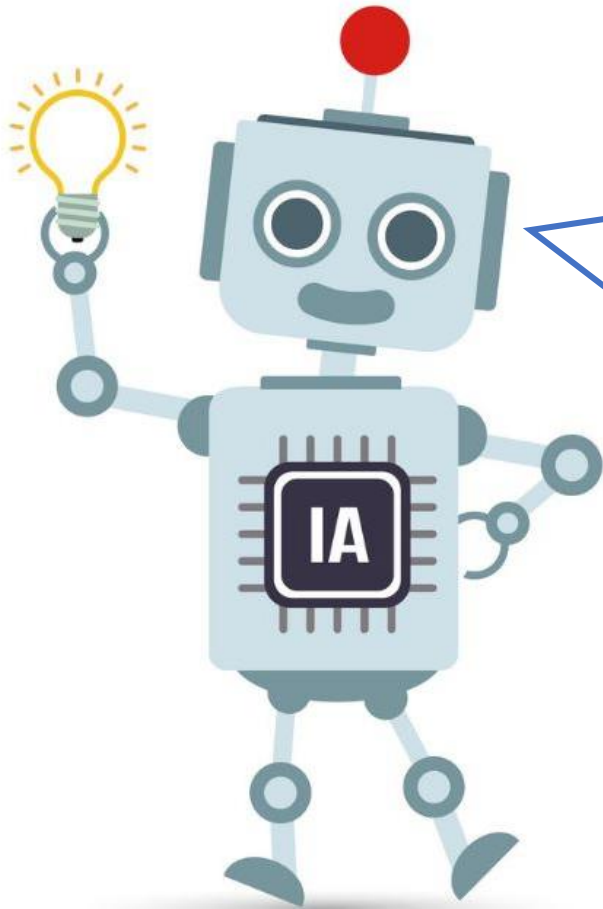


# 오늘 배운 내용

- **일상 생활의 많은 현상들은 선형적인 성격을 가진다.**
  - 선형 회귀 : 선형적인 관계에 적용하는 대표적인 기계학습이론이다.
- **학습을 시킨다.**
  - 주어진 데이터를 학습시켜서 가장 합리적인 '직선의 방정식'을 찾아낸다.
  - $H(x) = W * x + b$
- **(예측 값 - 실제 값)<sup>2</sup>의 평균으로 비용을 계산한다.**
  - 비용 함수 :  $Cost(W, b) = \frac{1}{m} \sum_{i=1}^m (H(x_i) - y_i)^2$
  - 학습은 경사 하강을 이용하여 비용을 최소화하는 방향으로 진행한다.
- **학습속도(Learning rate)는 프로그래머가 적절하게 정한다.**

# 다음 시간에 할 일

• SW중심대학



다음 시간 준비 사항 :

- 프로젝트팀 결성

다음 시간에 만나요~~~~~