# 이론과 실습, 두마리 토끼를 모두 잡는 세상에서 제일 쉬운 머신러닝 수업

기본모델(Regression, SVM, Decision Tree, Naive Bayes)부터
최신모델(Xgboost 등)까지


# ML Introduction

*Hypothesis, Cost, Optimization, Regularization, Cross Validation, Epoch & Batch*

# 01

# Data Science

*"DS cover computer science, mathematics, statistics, machine learning etc"*

❖ Data Science Competency

**Mathematic**

- Linear Algebra
- Calculus
- Optimization
- Probability

**Programming**

- Python, R, C etc.
- OS
- DB
- Docker, VM etc.

**Data Analysis**

- Probability
- Test & Estimate
- Regression etc.
- ML

**Domain Competency**

- Data -Understanding
- Preprocessing
- Feature -engineering

02

# What Machine Learning?

*"Not explicitily programmed & With task, performance measure, experience"*

"Field of study that gives computers the ability to learn without being explicitly programmed" Arthur Samuel (1959)

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" – T.Michell(1997)

Example1: A program for Go

T: Playing Go
P: Win or Lose
E: The number of Games

Example2: Spam mail Detection

T: Classifying emails as spam or ham
P: spam or ham
E: The number of classified emails

02

# What Machine Learning?

*"Not explicitly programmed & with T, P, E"*

❖ 인공지능 방법론

### 규칙기반(Rule-based)

어떤 입력이 들어오면 어떤 출력이 나오는지를 결정하는 규칙 혹은 알고 리즘을 사람이 미리 만들어 놓는 방법

### 학습기반(Training-based)

규칙을 사람이 만드는 것이 아니라 대량의 데이터를 컴퓨터에게 보여줌으 로써 스스로 규칙을 만들게 하는 방법

02

# What Machine Learning?

*"Not explicitly programmed & with T, P, E"*

## ❖ A.I. vs ML vs DL

**- 인공지능**

외부 관찰자에게 인간처럼 스마트하게 소프트웨어를 작동시키는 폭넓은 방법, 알고리즘 및 기술

머신러닝, 컴퓨터 비전, 자연어 처리, 로봇 공학 및 그와 관련된 모든 주제를 포괄하는 개념

**- 머신러닝**

더 많은 데이터 축적을 통해 성능을 개선할 수 있도록 하는 다양한 알고리즘과 방법론

신경망, 서포트 벡터 머신, 결정 트리, 베이지안 신뢰 네트워크, k 최근접 이웃, 자기 조직화 지도, 사례 기반 추론,

인스턴스 기반 학습, 은닉 마르코프 모델, 회귀 기법

**- 딥 러닝**

신경망(Neural Network)을 부르는 다른 이름

여러 개의 히든 레이어를 통해 깊게 학습한다고 해서 붙여진 이름

03

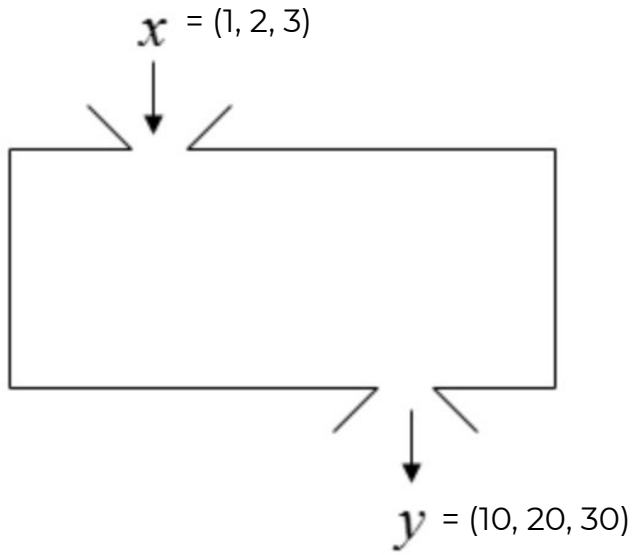
# Classification of Machine Learning

*"Supervised Learning & Unsupervised Learning"*

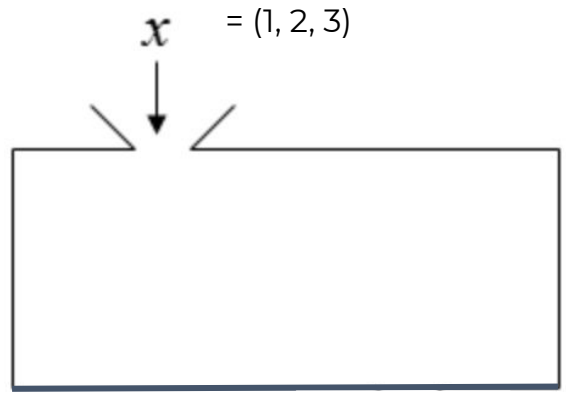
❖ Standard of Classification

DATA!

〈 Supervised Learning 〉




〈 Unsupervised Learning 〉



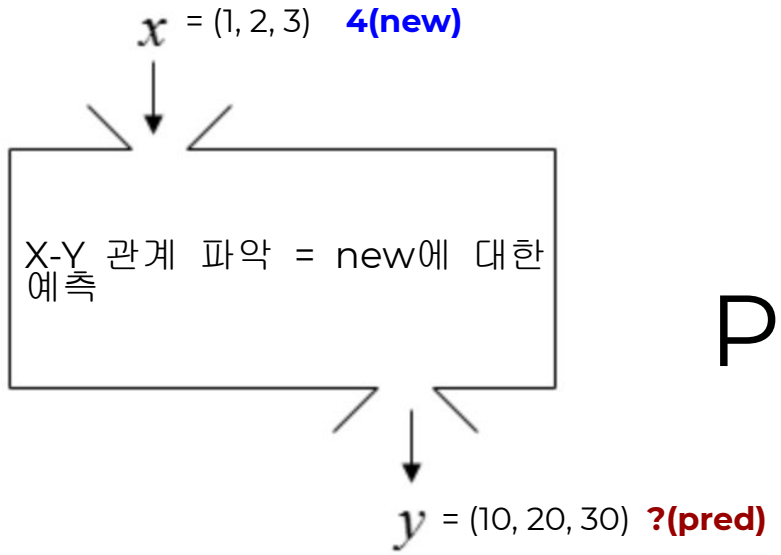
03

# Classification of Machine Learning

*"Supervised Learning & Unsupervised Learning"*

❖ Standard of Classification

〈 Supervised Learning 〉

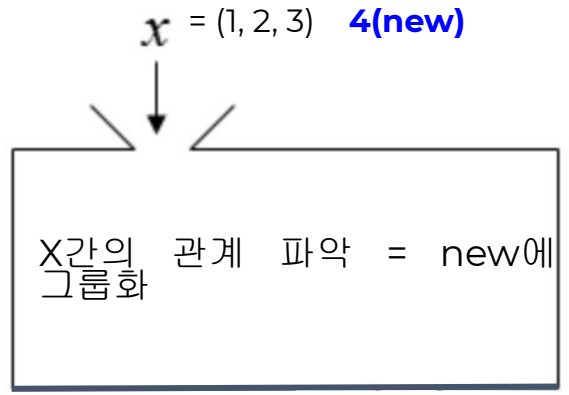



DATA!

||

PURPOSE!

〈 Unsupervised Learning 〉



04

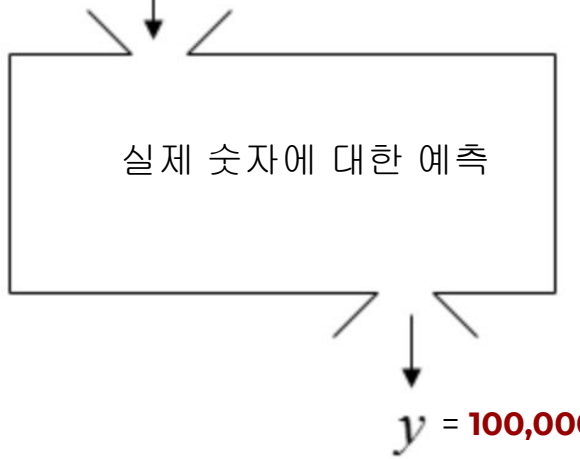
# Classification of Supervised Learning

*"Regression & Classification"*

❖ Standard of Classification

〈 Regression 〉

$x$ = Info_house_1, Info_house_2




$y$ = **100,000,000, 150,000,000**

y data!

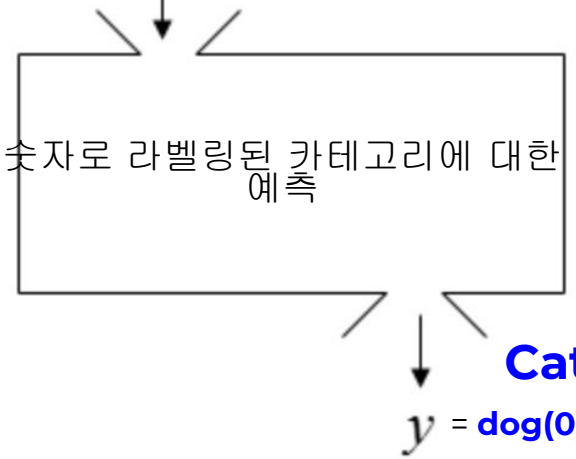
||

Target Value!

**Real Number**

〈 Classification 〉

$x$ = Picture_1, Picture_2




**Category Number**

$y$ = **dog(0), cat(1)**

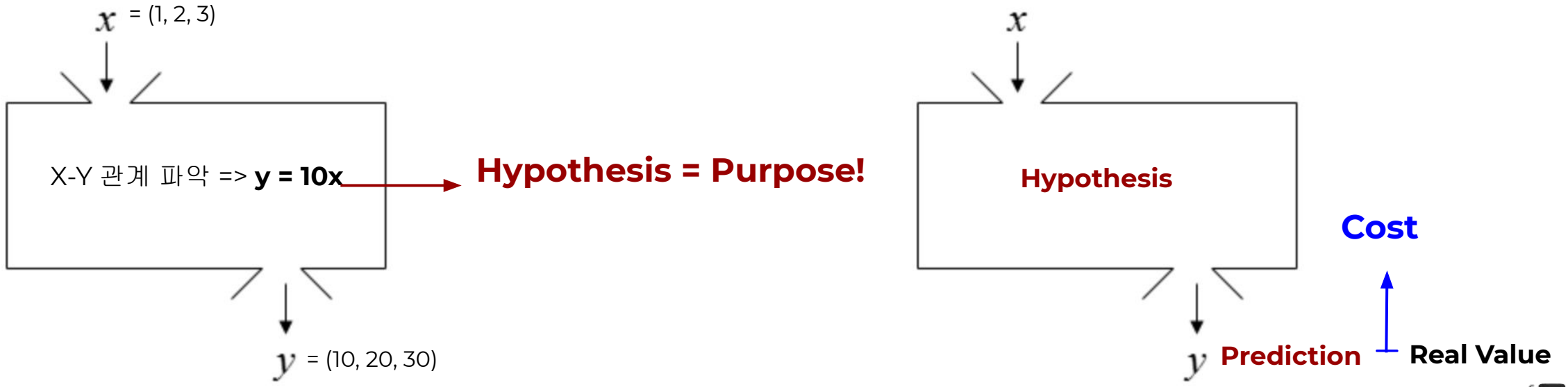
05

# How Machine Learning?

*"Hypothesis & Cost"*

❖ 2 essential function

〈 Supervised Learning 〉



05

# How Machine Learning?

*"Hypothesis & Cost"*

❖ 동의어 찾기!

**Purpose! = 예측 > X-Y관계파악 > Hypothesis 찾기!**

05

# How Machine Learning?

*"Hypothesis & Cost"*

❖ 동의어 찾기!

Purpose! = 정확한 ✓ 예측 > 정확한 ✓ X-Y관계파악 > 정확한 ✓ Hypothesis 찾기!

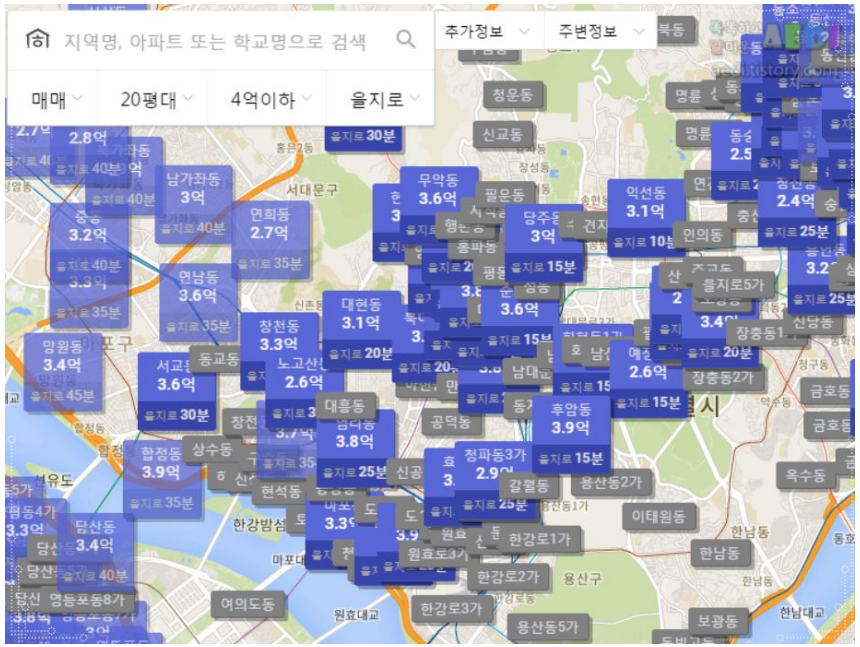
How 정확한 ?

05

# How Machine Learning?

*“Hypothesis & Cost”*

Step 1. 우선 “ 첫번째 Hypothesis ” 를 결정한다.

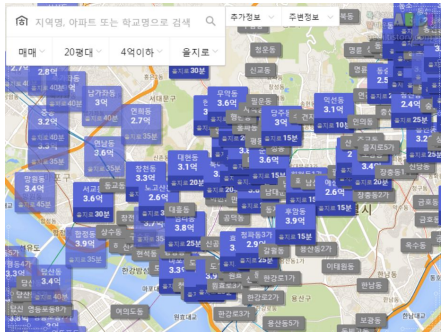


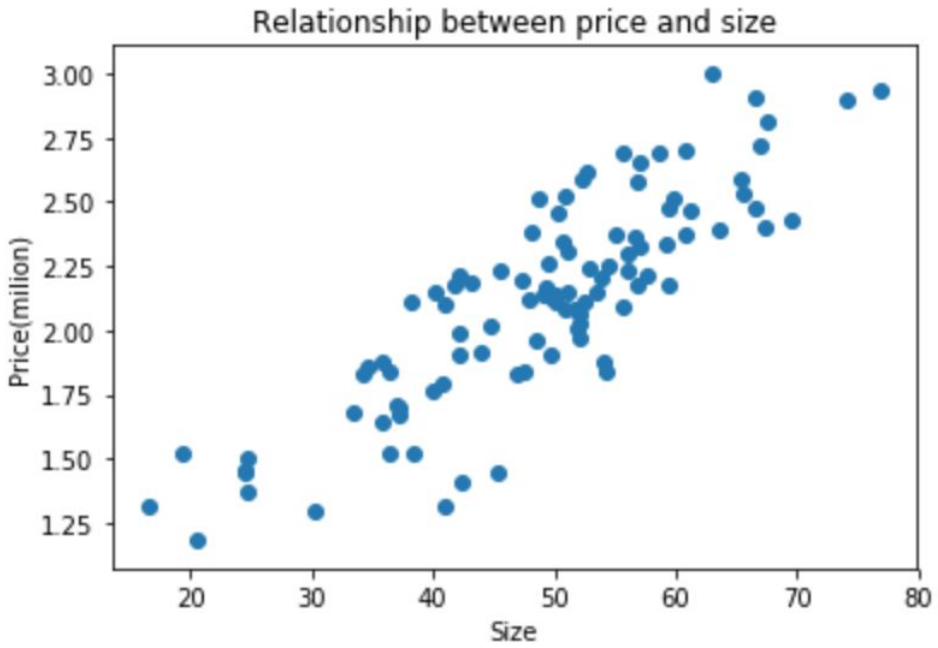
05

# How Machine Learning?

*"Hypothesis & Cost"*

Step 1. 우선 " 첫번째 Hypothesis " 를 결정한다.



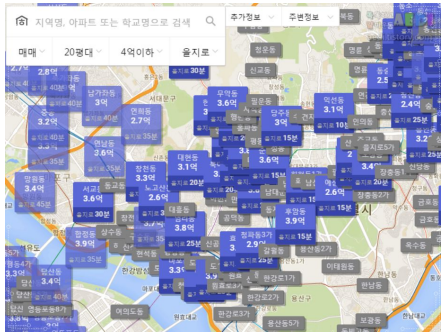


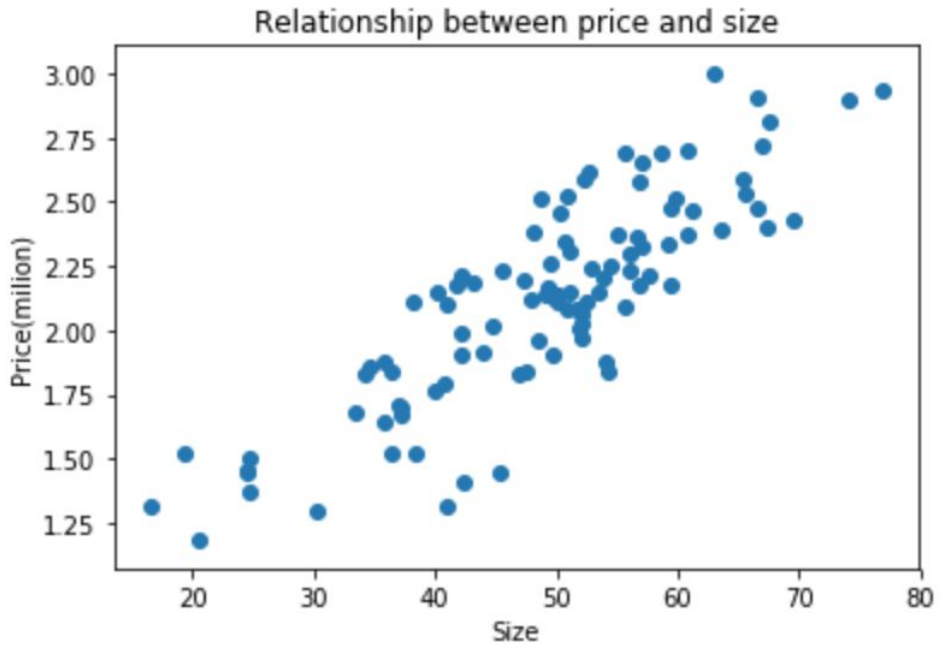
05

# How Machine Learning?

*“Hypothesis & Cost”*

Step 1. 우선 “ 첫번째 Hypothesis ” 를 결정한다.





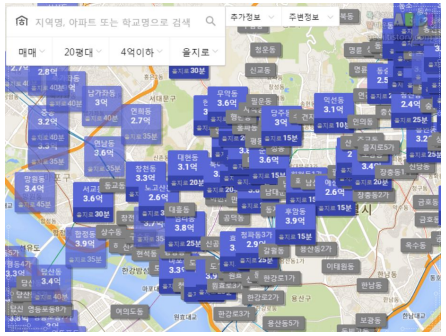

Size와 Price의 관계를
최대한 단순하게(직선으로)
찾는다면?

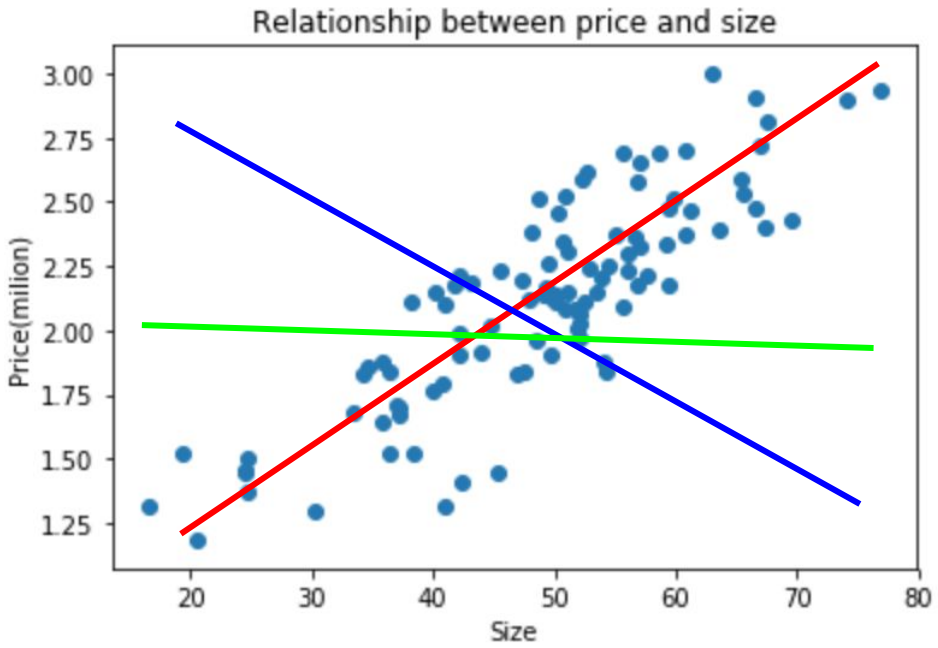
05

# How Machine Learning?

*"Hypothesis & Cost"*

Step 1. 우선 " 첫번째 Hypothesis " 를 결정한다.





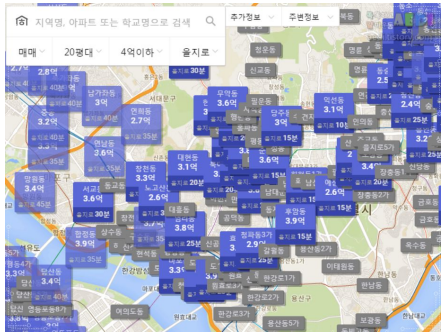

1번. 빨간선
2번. 파란선
3번. 초록선

05

# How Machine Learning?

*"Hypothesis & Cost"*

Step 1. 우선 " 첫번째 Hypothesis " 를 결정한다.




당연히 빨간선을 선택할 것이다.
그러나 기계(Machine)는
이러한 직관이 없다.

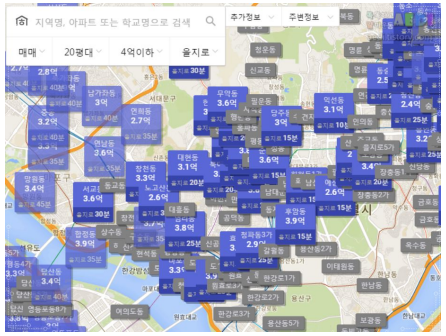
그럼.. 어떻게..?

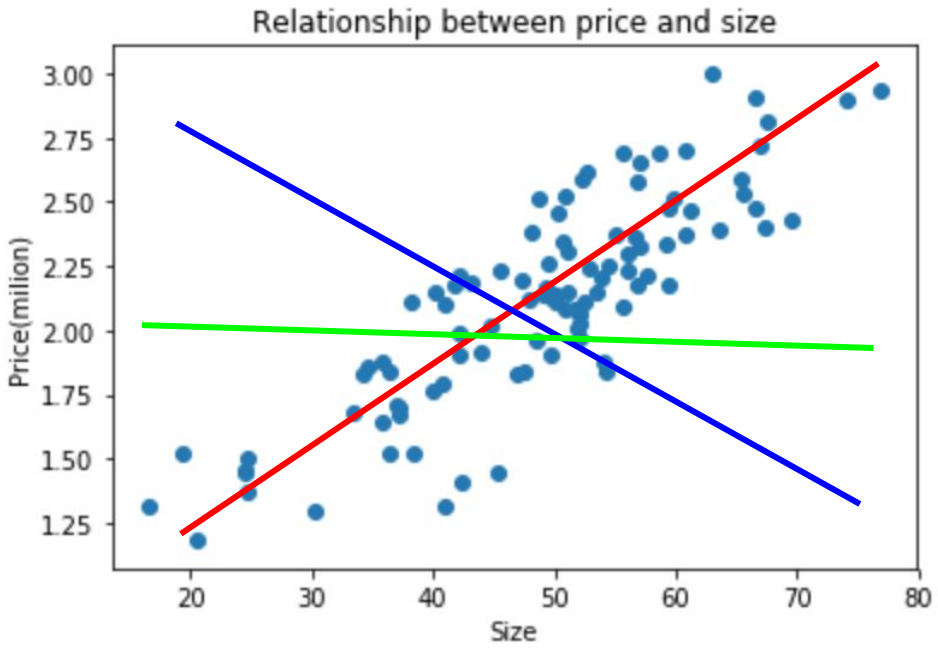
05

# How Machine Learning?

*"Hypothesis & Cost"*

Step 1. 우선 " 첫번째 Hypothesis " 를 결정한다.





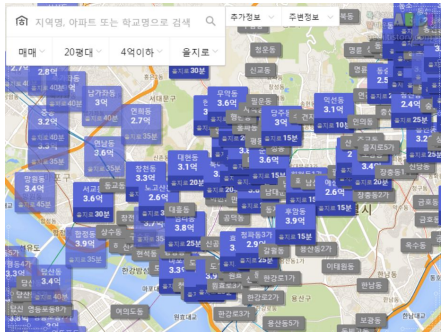

1번. 빨간선
2번. 파란선
3번. 초록선

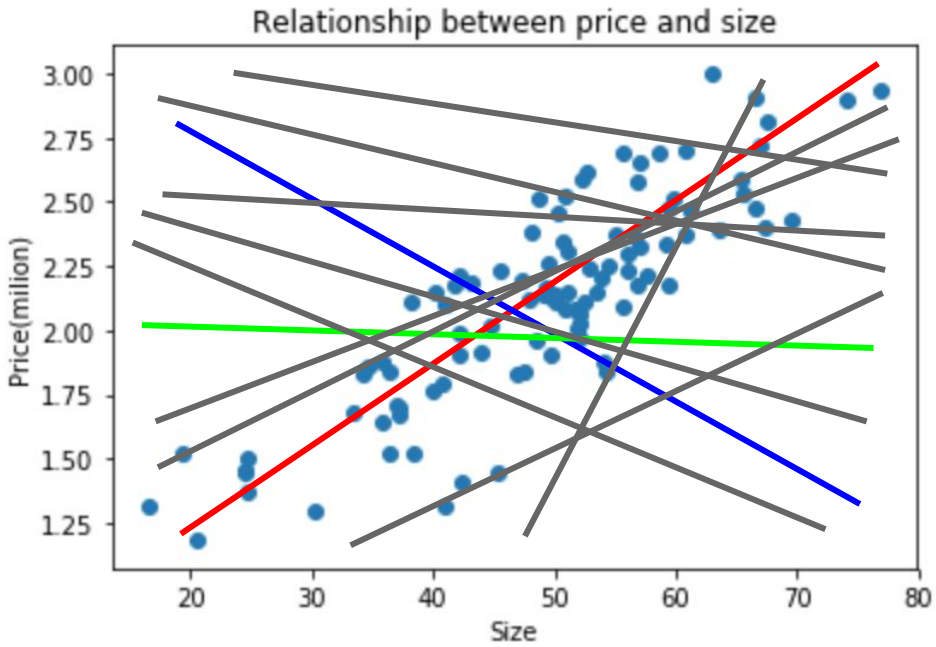
05

# How Machine Learning?

*"Hypothesis & Cost"*

Step 1. 우선 " 첫번째 Hypothesis " 를 결정한다.





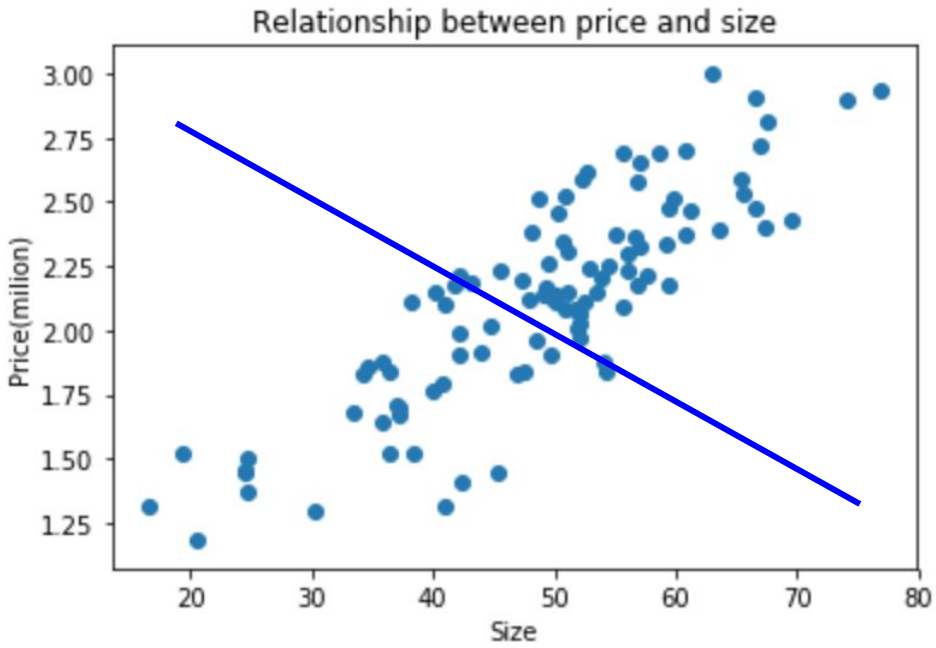

1번. 빨간선
2번. 파란선
3번. 초록선
4번. …
…
1003번. …
…

05

# How Machine Learning?

*“Hypothesis & Cost”*

Step 1. **우선 “ 첫번째 Hypothesis ” 를 결정한다.**



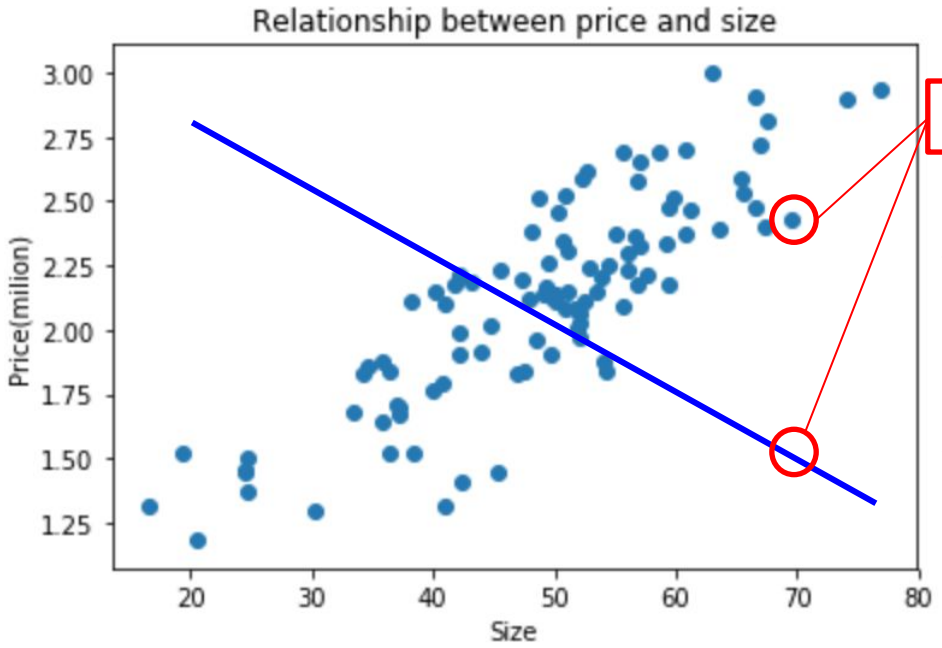

눈감고 아무거나..
혹은 좀 더 영리하게..
어쨌든 여러가지 중 하나
ex) 2번. 파란선

05

# How Machine Learning?

*"Hypothesis & Cost"*

Step 2. 결정된 첫번째 Hypothesis에 대한 " 첫번째 Cost " 를 확인한다.




예측값과 실제값의 차이 = **"err"** 라고 하자.

모든 데이터에 대해 err값을 계산해 평균 값
= "Cost"라고 한다.

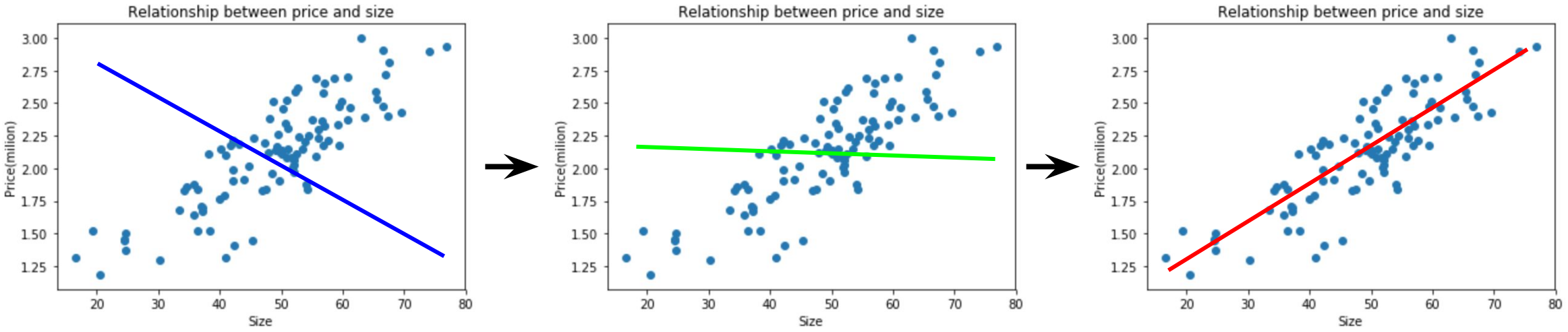
(단, 그냥 합하면 +,- 상쇄되니까
제곱하든, 절대값하든 상쇄요소 제거 후, 평균)

05

# How Machine Learning?

*"Hypothesis & Cost"*

Step 3. " Cost가 낮아지는 방향 "으로 Hypothesis를 Update한다.

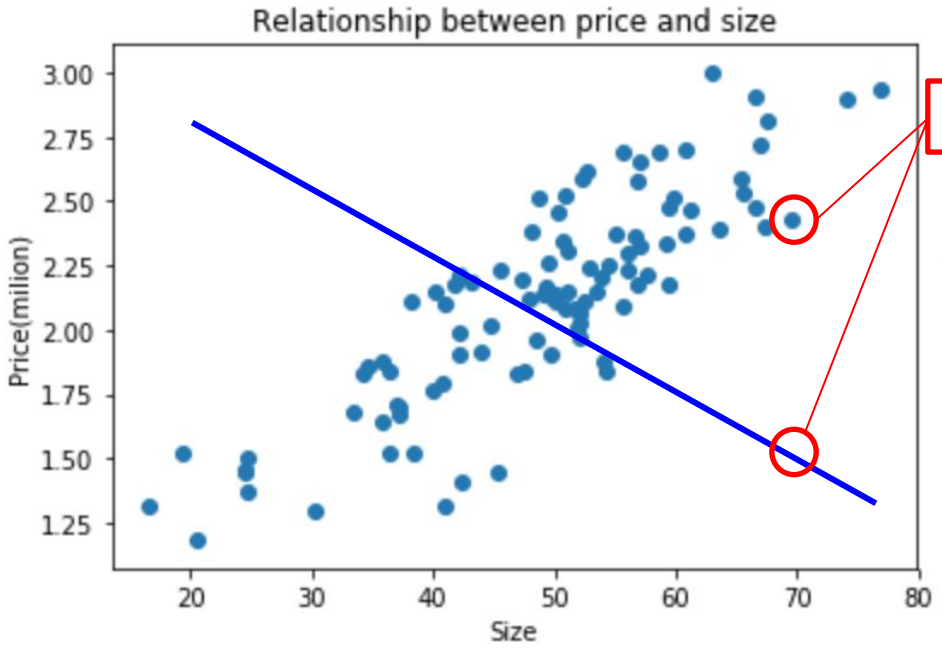


06

# Identity of Cost

*"Hypothesis & Cost"*

Step 2. 결정된 첫번째 Hypothesis에 대한 " 첫번째 Cost " 를 확인한다.




예측값과 실제값의 차이 = **"err"** 라고 하자.

모든 데이터에 대해 err값을 계산해 평균 값
= "Cost"라고 한다.

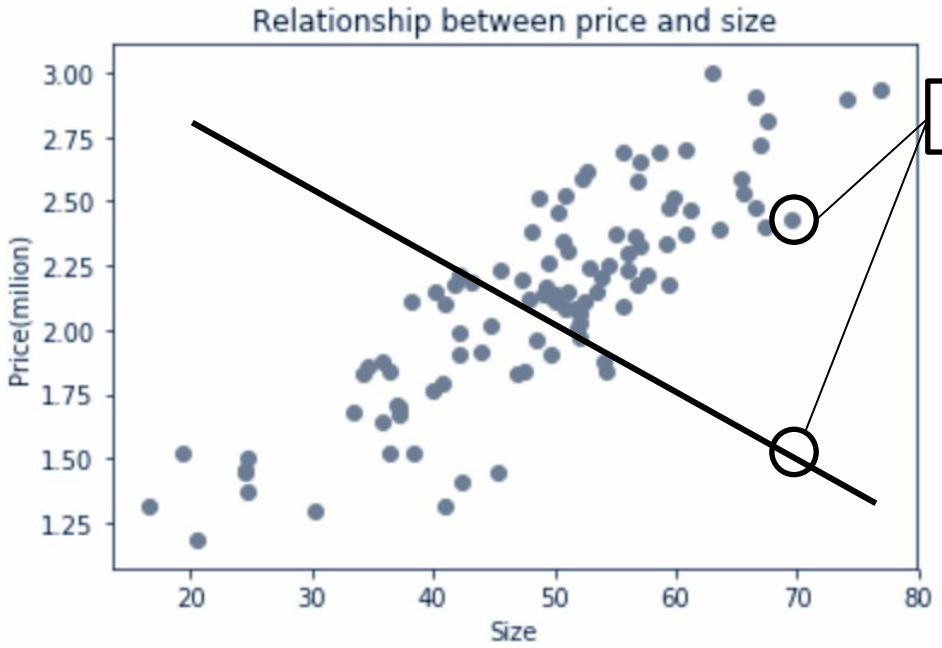
(단, 그냥 합하면 +,- 상쇄되니까
제곱하든, 절대값하든 상쇄요소 제거 후, 평균)

06

# Identity of Cost

*“Hypothesis & Cost”*

**결정된 첫번째 Hypothesis에 대한 “ 첫번째 Cost ” 를 확인한다?**




예측값과 실제값의 차이 = **“err”** 라고 하자.

모든 데이터에 대해 err값을 계산해 평균 값
= “Cost”라고 한다.

(단, 그냥 합하면 +,- 상쇄되니까
제곱하든, 절대값하든 상쇄요소 제거 후, 평균)

06

# Identity of Cost

*"Hypothesis & Cost"*

**결정된 첫번째 Hypothesis에 대한 " 첫번째 Cost " 를 확인한다?**


예측값과 실제값의 차이 = **"err"** 라고 하자.

모든 데이터에 대해 err값을 계산해 평균 값
= "Cost"라고 한다.

(단, 그냥 합하면 +,- 상쇄되니까
제곱하든, 절대값하든 상쇄요소 제거 후, 평균)

06

# Identity of Cost

*“Hypothesis & Cost”*

**결정된 첫번째 Hypothesis에 대한 “ 첫번째 Cost ” 를 확인한다?**


**“err”** 가 계산이 된다. = Cost가 계산이 된다.

06

# Identity of Cost

*“Hypothesis & Cost”*

**결정된 첫번째 Hypothesis에 대한 “ 첫번째 Cost ” 를 확인한다?**

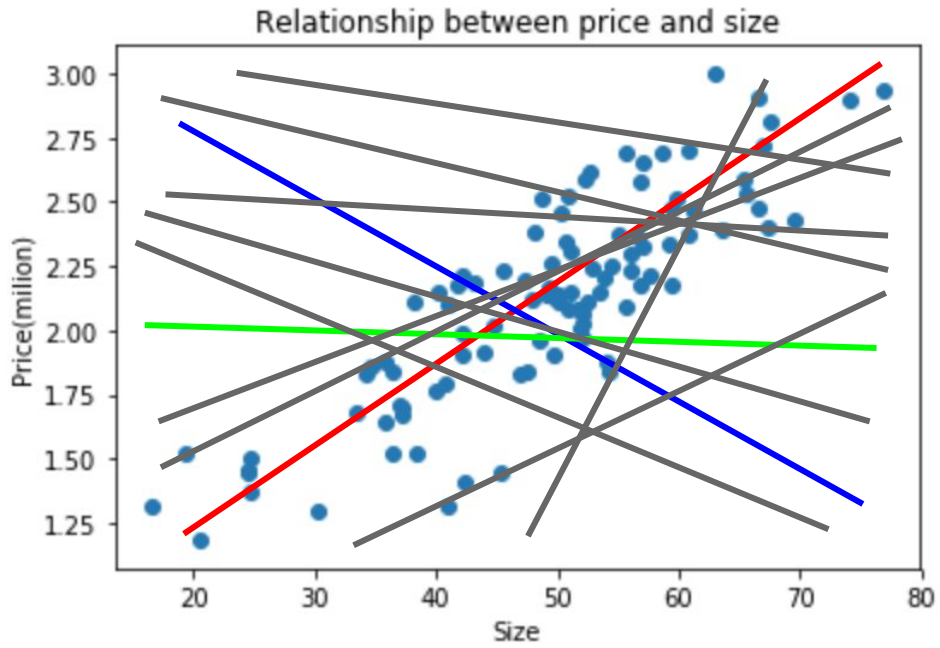

**“err”** 가 계산이 된다. = Cost가 계산이 된다.

06

# Identity of Cost

*"Hypothesis & Cost"*

**결정된 첫번째 Hypothesis에 대한 " 첫번째 Cost " 를 확인한다?**




1번. 빨간선
2번. 파란선
3번. 초록선
4번. …
…
1003번. …
…

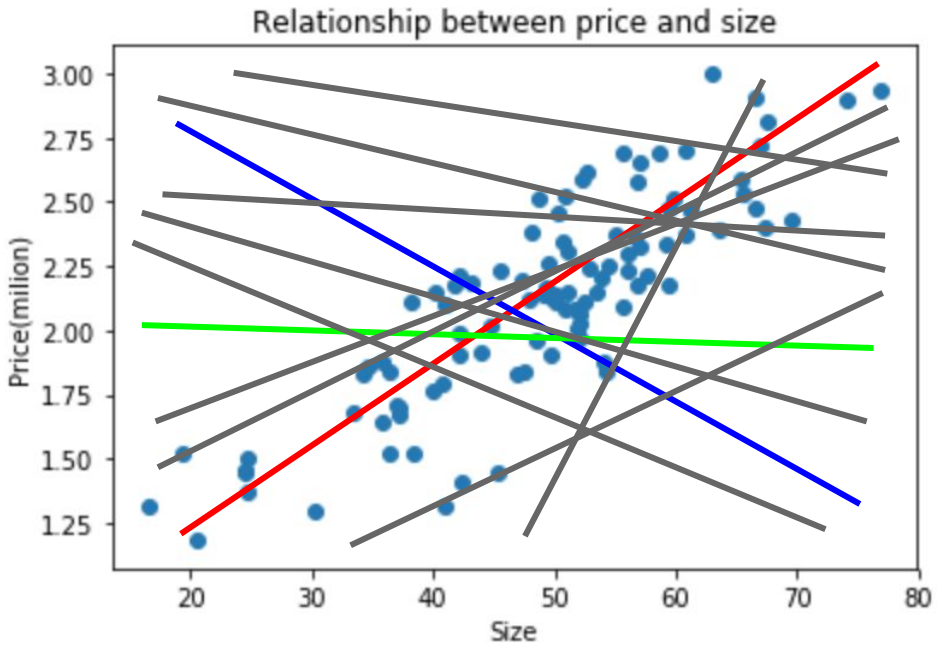
**모든 Hypothesis는 자신의 Cost를 갖고 있다!**

06

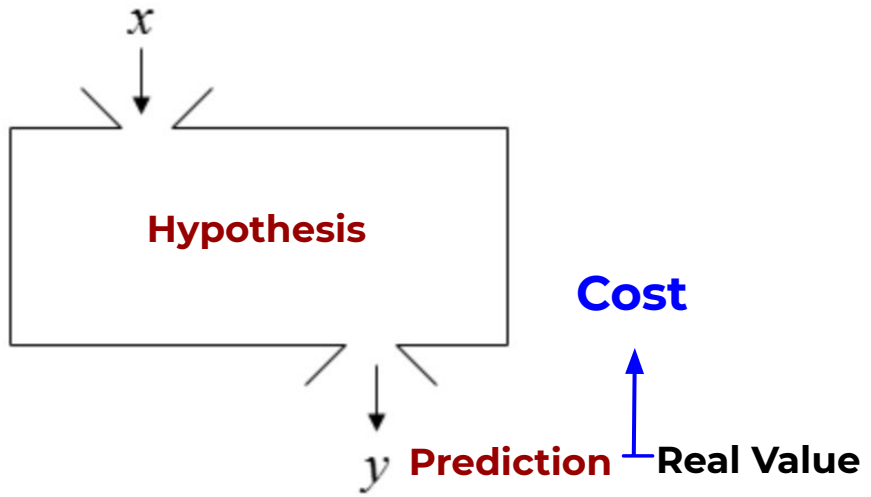
# Identity of Cost

*"Hypothesis & Cost"*

결정된 첫번째 Hypothesis에 대한 " 첫번째 Cost " 를 확인한다?




1번. 빨간선
2번. 파란선
3번. 초록선
4번. …
…
1003번. …
…



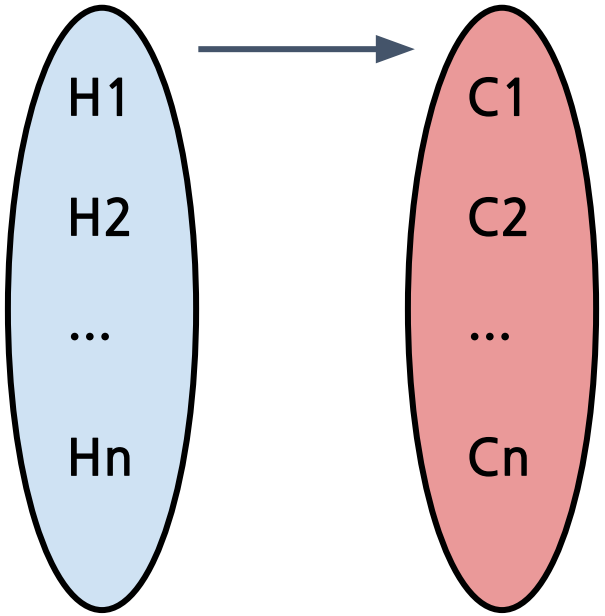

모든 Hypothesis는 자신의 Cost를 갖고 있다!

06

# Identity of Cost

*“Hypothesis & Cost”*

**모든 Hypothesis는 자신의 Cost를 갖고 있다!**

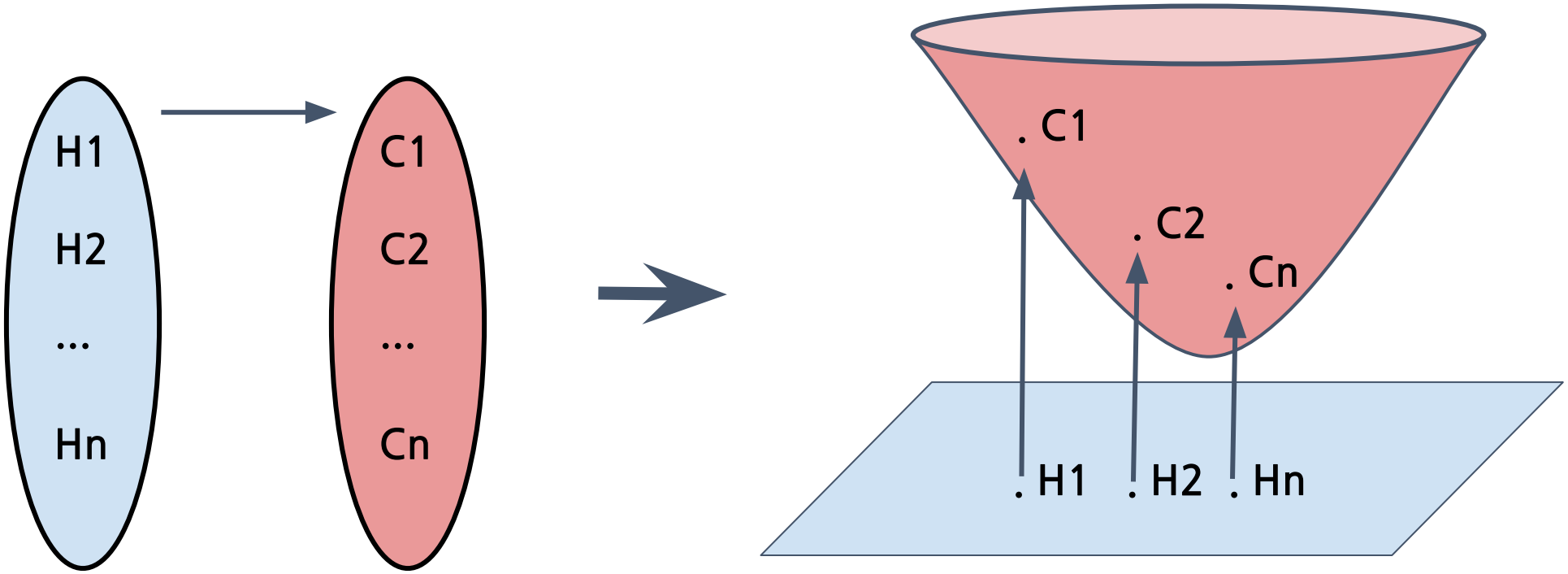


06

# Identity of Cost

*"Hypothesis & Cost"*

**모든 Hypothesis는 자신의 Cost를 갖고 있다!**



06

# Identity of Cost

*"Hypothesis & Cost"*

❖ 동의어 찾기!

정확한 ✓ 정확한 ✓ 정확한 ✓

**Purpose! = 예측 > X-Y관계파악 > Hypothesis 찾기!**

How 정확한 ?

06

# Identity of Cost

*"Hypothesis & Cost"*

❖ 동의어 찾기!

정확한 ✓ 정확한 ✓ 정확한 ✓

**Purpose! = 예측 > X-Y관계파악 > Hypothesis 찾기!**

Cost가 낮을 때!

06

# Identity of Cost

*"Hypothesis & Cost"*

❖ 동의어 찾기!

정확한 ✓ 정확한 ✓ cost가 낮은 ✓

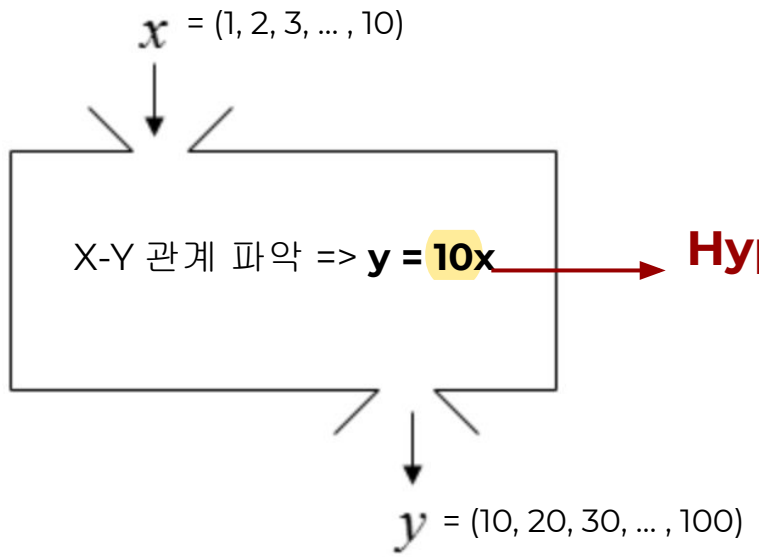
**Purpose! = 예측 > X-Y관계파악 > Hypothesis 찾기!**

Cost가 낮다! = Hypothesis가 정확하다!

07

# Identity of Hypothesis

*"Weight"*

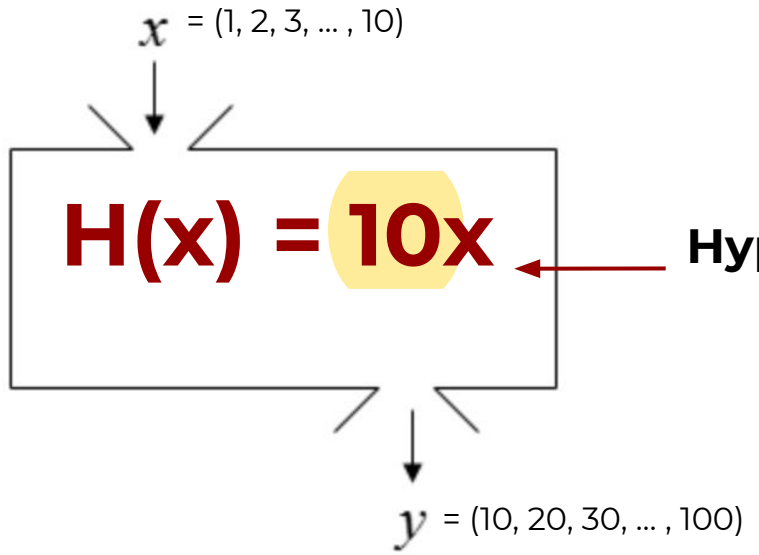



**Hypothesis = Purpose!**

07

# Identity of Hypothesis

*"Weight"*



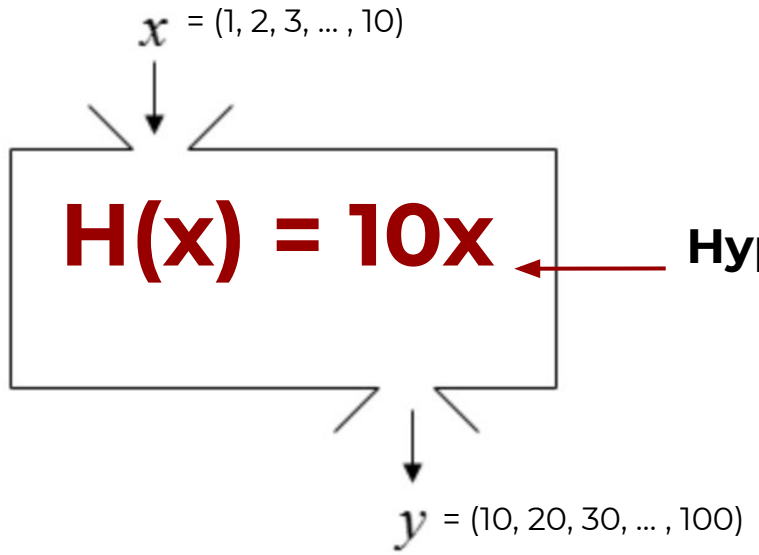

**Hypothesis = X - Y 관계**

**Hypothesis 를 결정했다? = "10"을 결정했다.**

07

# Identity of Hypothesis

*“Weight”*




**Hypothesis = X - Y 관계**

**Hypothesis 를 결정했다? = “Weight”를 결정했다.**

07

# Identity of Hypothesis

*“Weight”*

Step 1. **우선 “ 첫번째 Hypothesis ” 를 결정한다. ⇒ 우선 “ 첫번째 Weights ” 를 결정한다.**


눈감고 아무거나..
혹은 좀 더 영리하게..
어쨌든 여러가지 중 하나
ex) 2번. 파란선 = **a**∗size + **b**

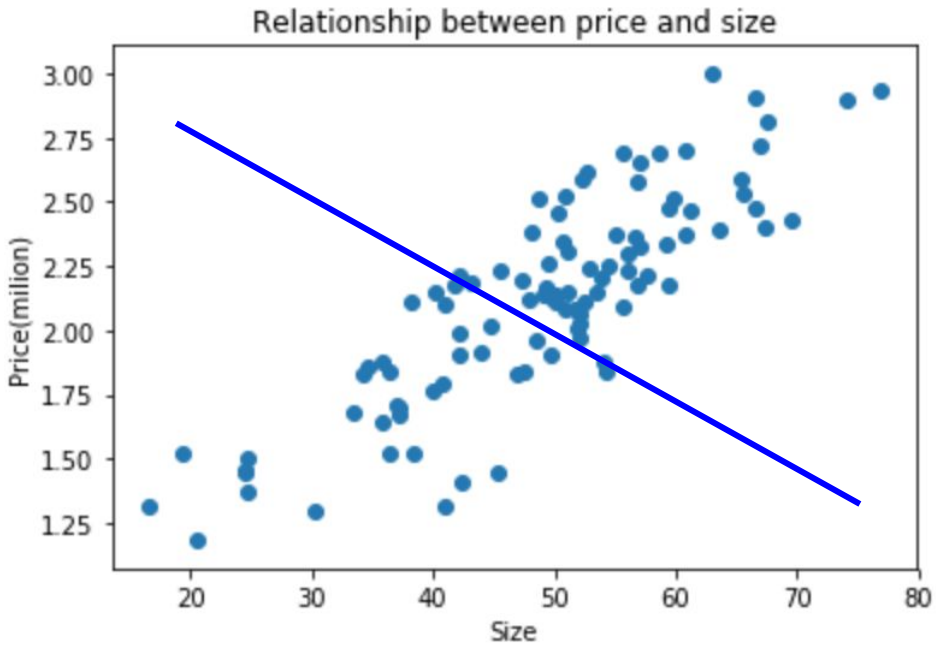
y = a∗x+b ⇒ a: , b:

07

# Identity of Hypothesis

*"Weight"*

Step 1. **우선 " 첫번째 Hypothesis " 를 결정한다. ⇒ 우선 " 첫번째 Weights " 를 결정한다.**




눈감고 아무거나..
혹은 좀 더 영리하게..
어쨌든 여러가지 중 하나
ex) 2번. 파란선 = **W1***size + **W0**

W1: , W0:

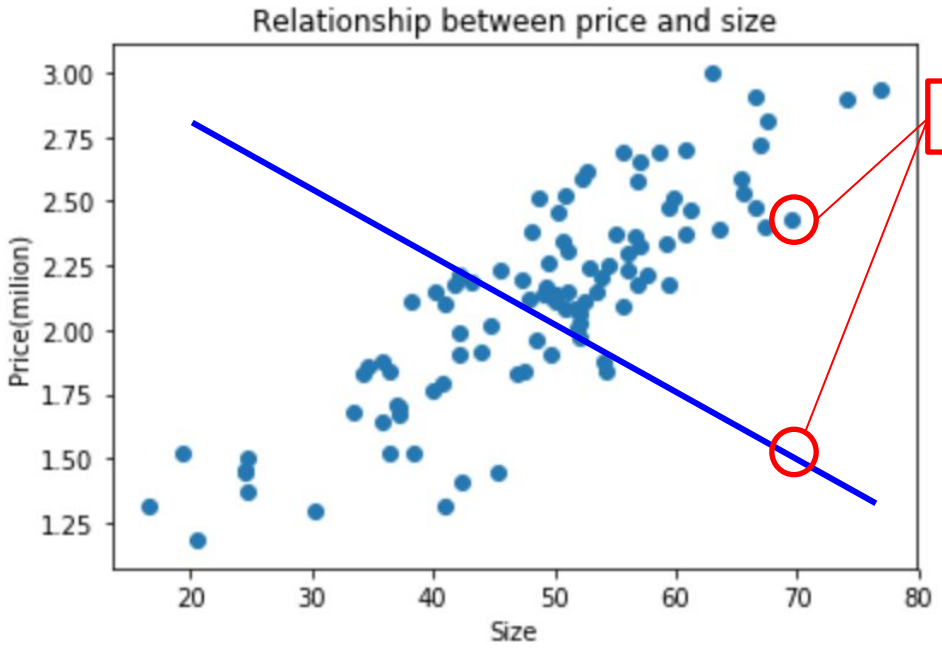
Ardino

07

# Identity of Hypothesis

*“Weight”*

Step 2. 결정된 “ **첫번째 Weights** ” 에 대한 “ 첫번째 Cost ” 를 확인한다.




예측값과 실제값의 차이 = **“err”** 라고 하자.

모든 데이터에 대해 err값을 계산해 평균 값
= “Cost”라고 한다.

(단, 그냥 합하면 +,- 상쇄되니까
제곱하든, 절대값하든 상쇄요소 제거 후, 평균)

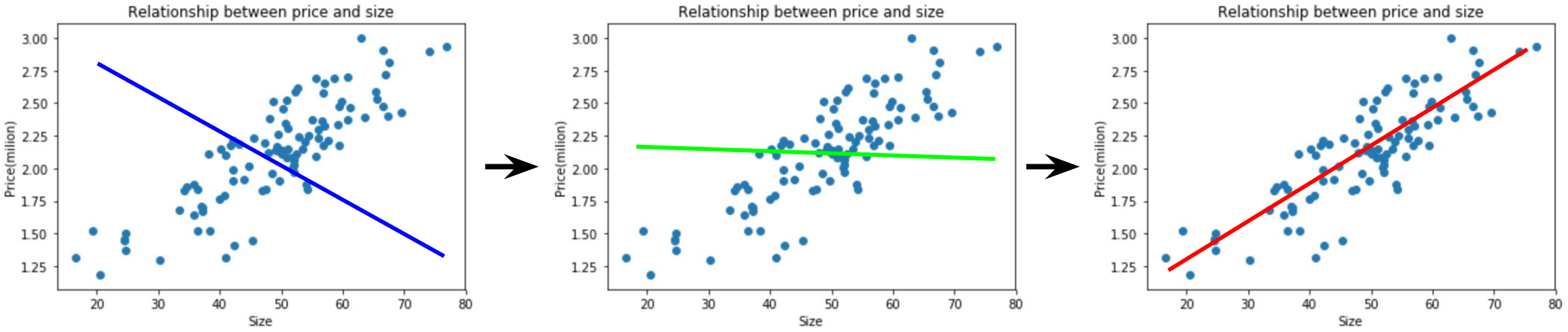
07

# Identity of Hypothesis

*"Weight"*

Step 3. " Cost가 낮아지는 방향 "으로 " **Weights** " 를 Update한다.

weight : 기울기, bios : Y절편

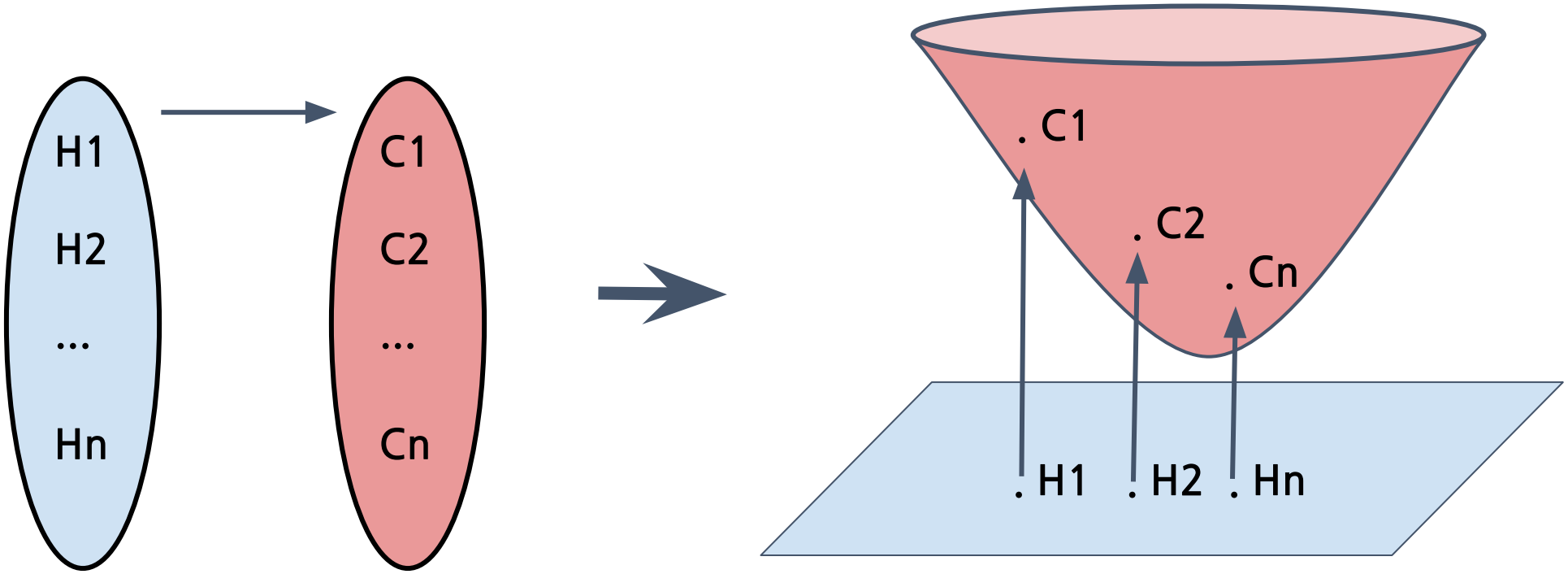


07

# Identity of Hypothesis

*"Weight"*

**모든 Hypothesis는 자신의 Cost를 갖고 있다!**

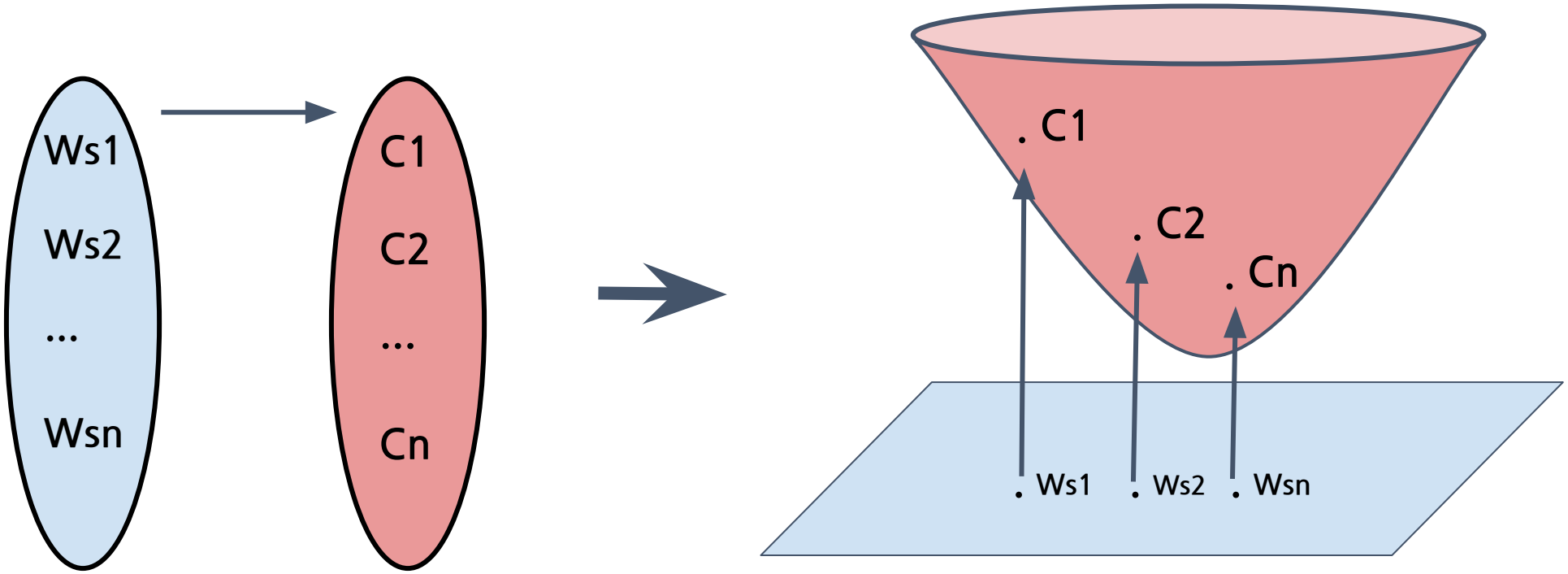


07

# Identity of Hypothesis

*"Weight"*

**모든 Weights는 자신의 Cost를 갖고 있다!**



07

# Identity of Hypothesis

*"Weight"*

❖ 동의어 찾기!

정확한 ✓ 정확한 ✓ cost가 낮은 ✓ cost가 낮은 ✓

Purpose! = 예측 = X-Y관계파악 = Hypothesis 찾기! = Weights 찾기

Cost가 낮다! = Hypothesis가 정확하다! = Weights가 정확하다!

07

# Identity of Hypothesis

*“Weight”*

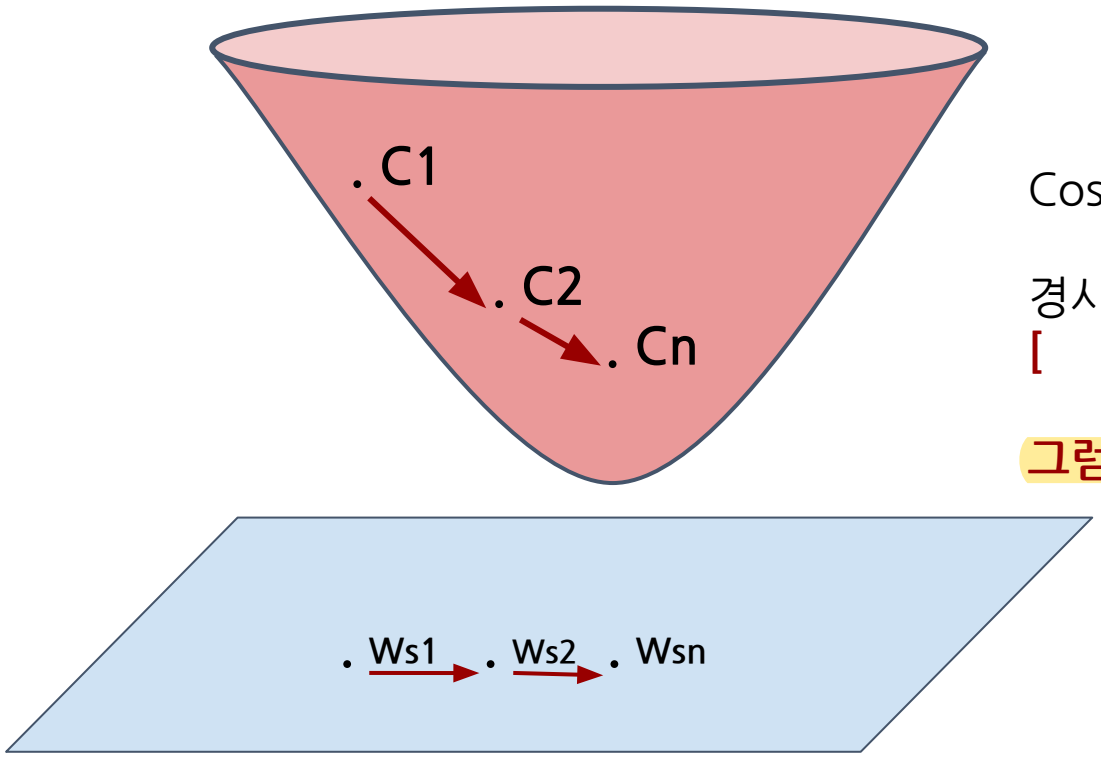
❖ 머신러닝을 한마디로!

**“Cost가 가장 낮을 때의 Weights를 찾는 과정!”**

08

# Optimization

*"Gradient Descent"*




Cost값 자체를 보는게 아니고,

경사를 보고, 경사가 완만해지는 방향으로
[ ]를 Update!

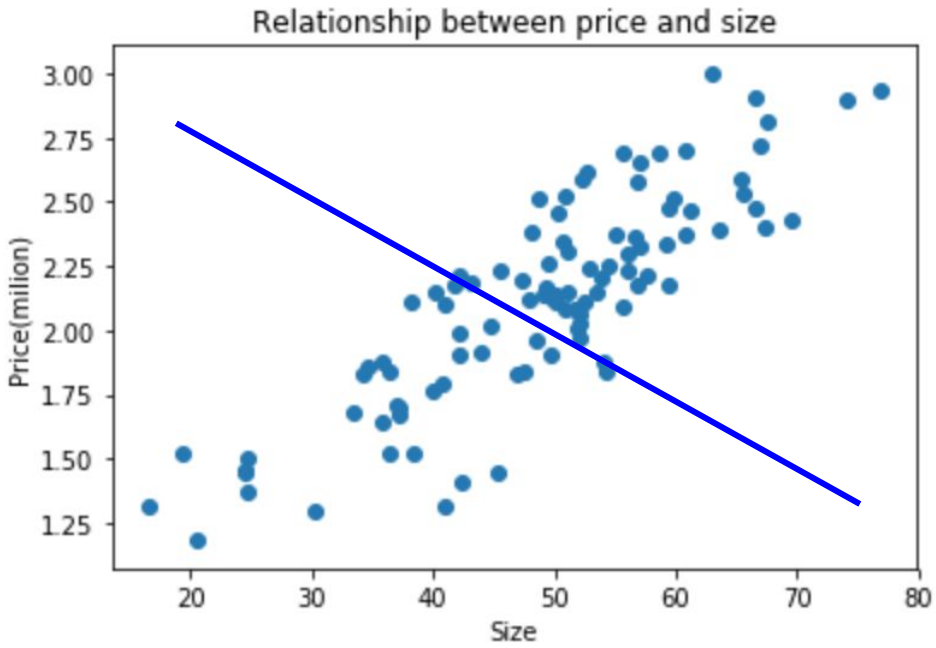
**그럼 결국 언제 Update를 멈출까? ⇒ 경사가 0일때!**

08

# Optimization

*"Local Minimum"*

Step 1. **우선 " 첫번째 Hypothesis " 를 결정한다. ⇒ 우선 " 첫번째 Weights " 를 결정한다.**

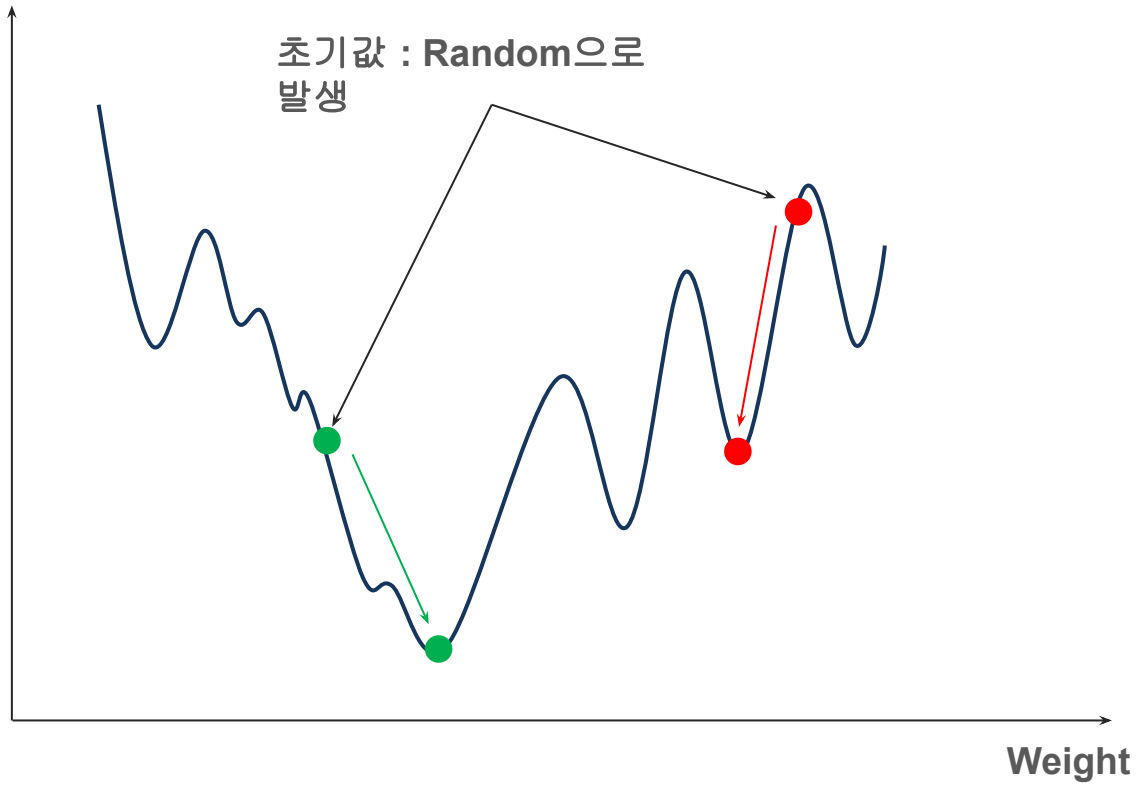



눈감고 아무거나..
혹은 좀 더 영리하게..
어쨌든 여러가지 중 하나
ex) 2번. 파란선 = **W1***size + **W0**

08

# Optimization

*"Local Minimum"*




■ SGD, Momentum, Adams …

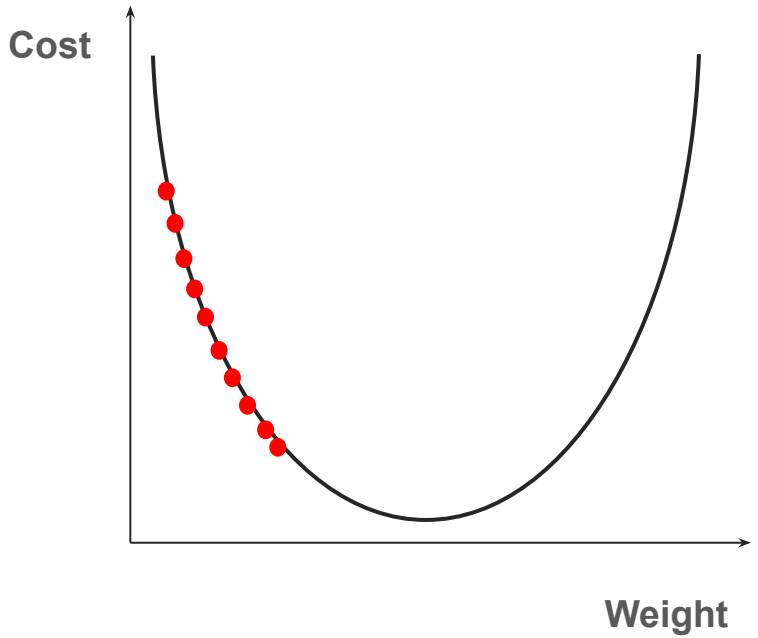
08

# Optimization

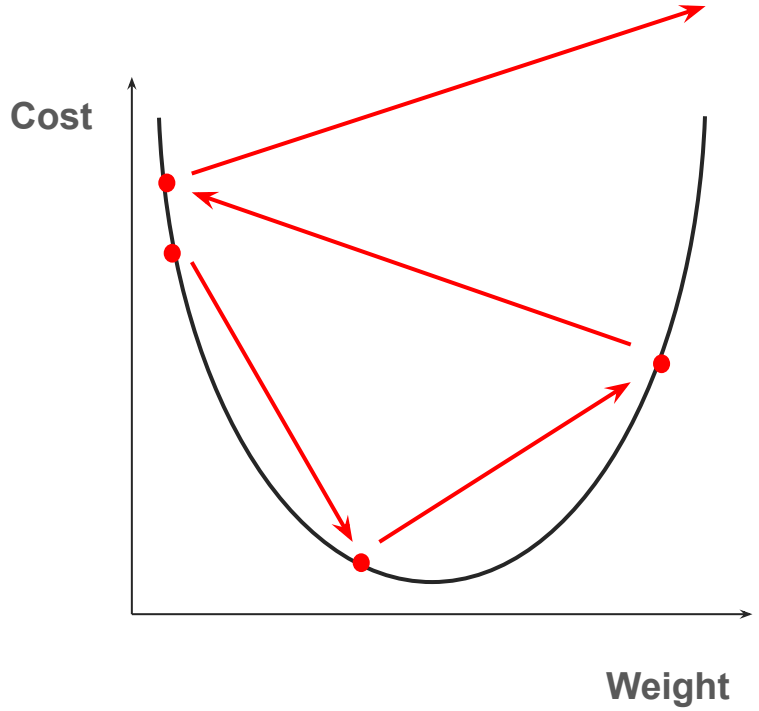
R

"*Learning ~~L~~ate*"

- Too Small



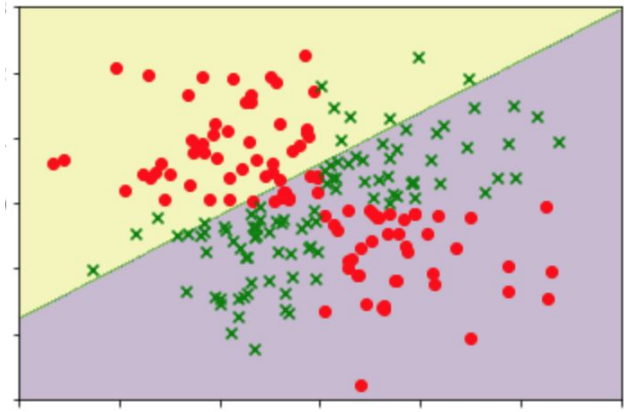

- Too Large



09

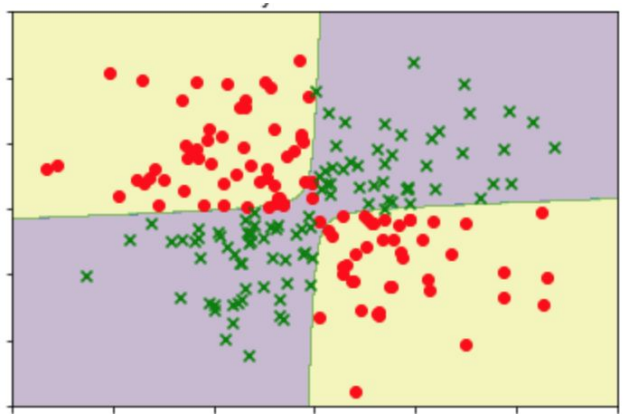
# Regularization

*"Overfitting"*



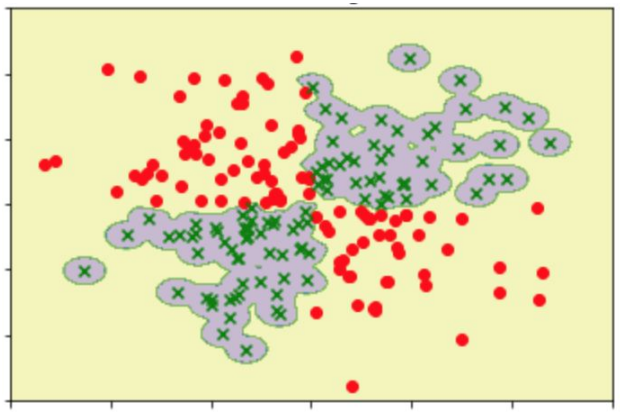
**Underfitting**

New data에 대한 예측을 잘 못한다.



**Fitting**

New data에 대한 예측을 잘 한다.



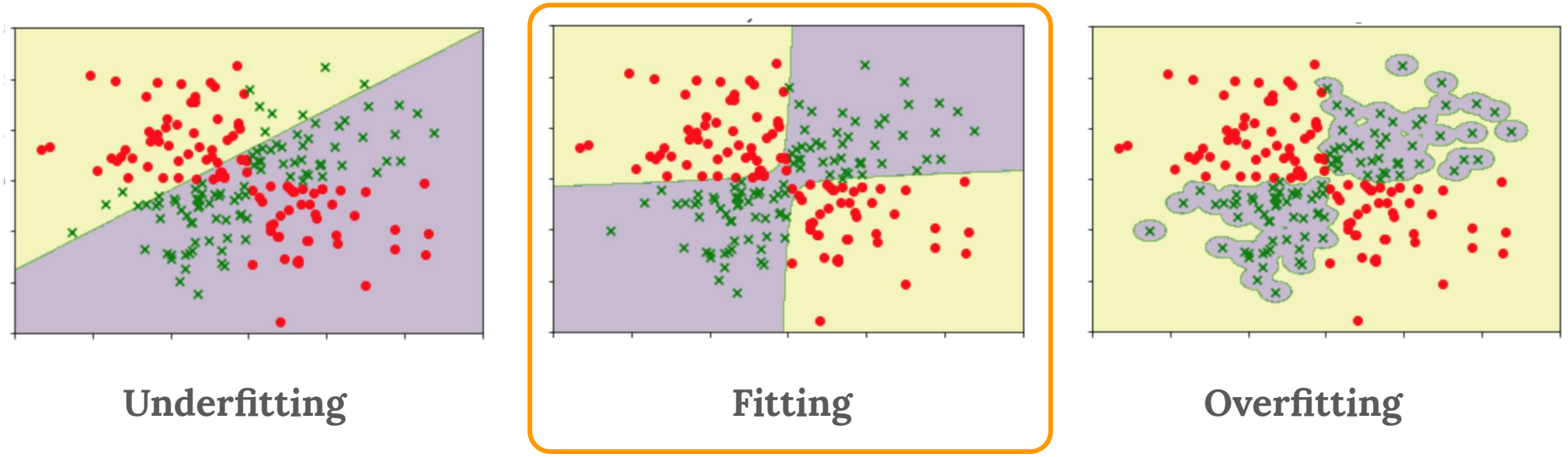
**Overfitting**

New data에 대한 예측을 잘 못한다.

09

# Regularization

"*Overfitting*"



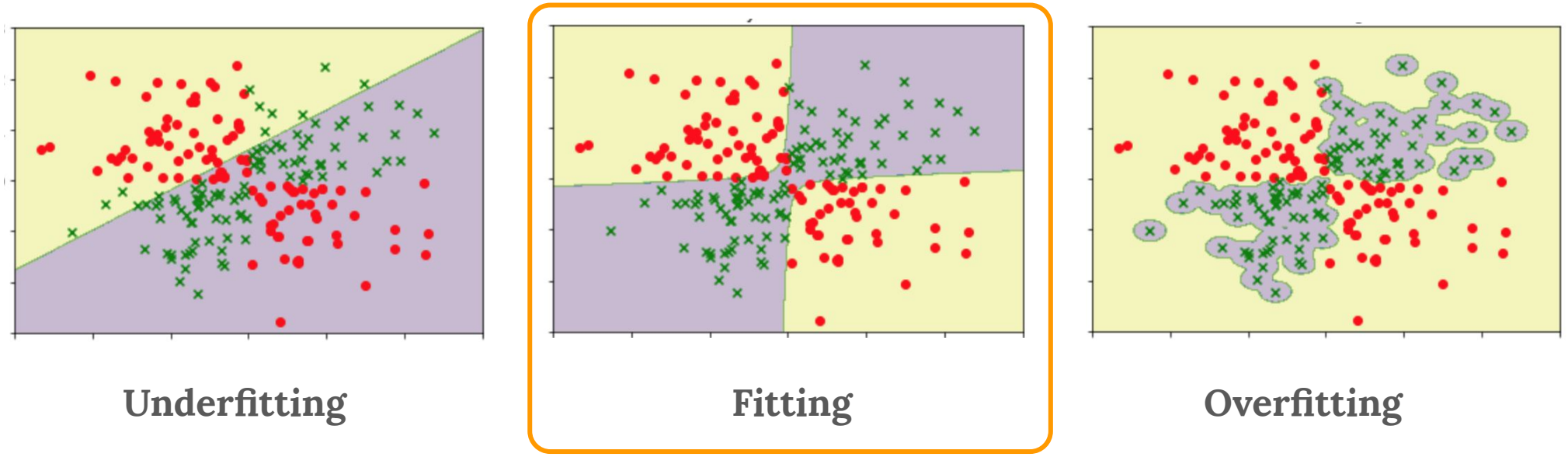
Underfitting | Fitting | Overfitting

결론! 오버하지 말고, 적당히!

09

# Regularization

*"Overfitting"*

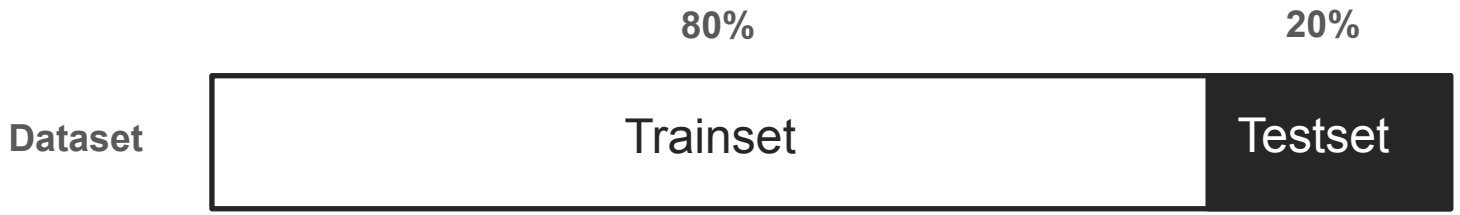


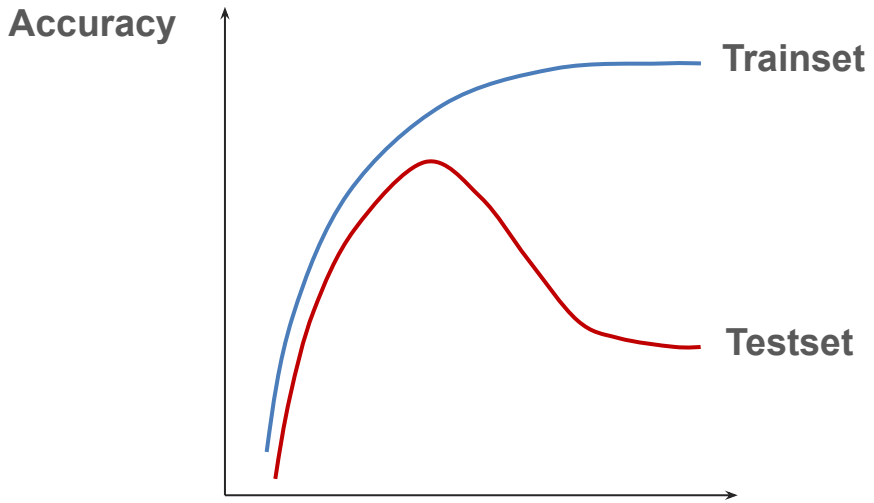
Underfitting | Fitting | Overfitting

적당히 학습하도록  규제화라고 부른다. 학습시킬 때 우리가 이 정도를 정해줄 수 있다!

# Cross Validation

*"Trainset vs Testset"*



# Cross Validation

*"Trainset vs Testset"*