기계학습의 원리, 능력과 한계

2016.3.

김 진 형

- 소프트웨어정책연구소 소장
- KAIST 전산학부 명예교수
- 국제패턴인식학회 Fellow
- 정보과학회 명예회장

소프트웨어 중심사회의 Think Tank SPRi Software Policy & Research Institute

결과는 종종 혁신적이지만
진화는 항상 점진적이다*

혁신적인 알파고, 딥러닝은
70년 동안의 인공지능기술 진화의 산물

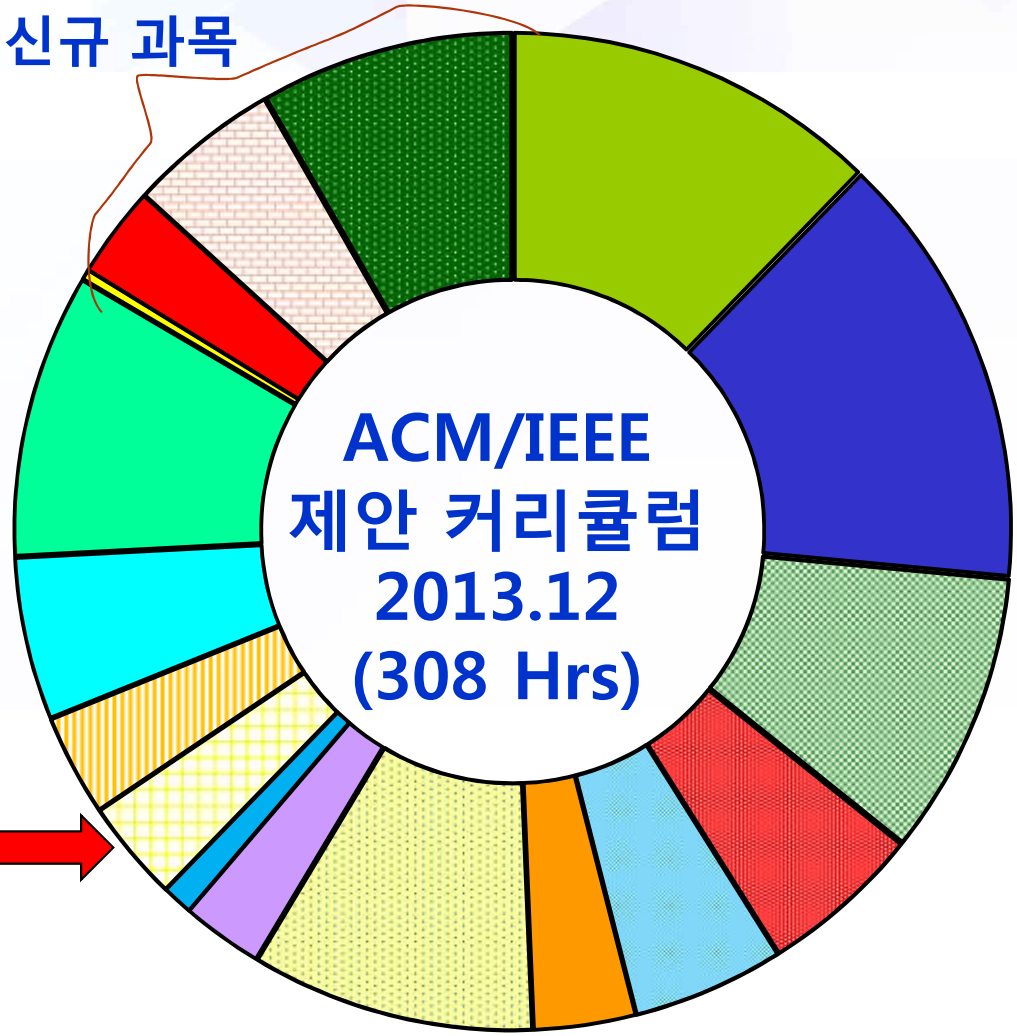
* 출처: “기술의 진화: 비유와 함의들”, 이관수(동국대 다르마칼리지 이관수 교수)에서

인공지능이란 ?

- 지능적 행동을 자동화 하기 위한 컴퓨터 과학의 한 분야
  - (Luger & Stubblefield, 1993)
- 현재 사람이 더 잘 하는 일을 컴퓨터가 하도록 하는 연구
  - (Rich & Knight, 1991)
- 컴퓨터를 좀 더 스마트하게 만들기
- 생각하는 컴퓨터 만들기

인공지능은 컴퓨터과학의 핵심

2013년 신규 과목



인공지능

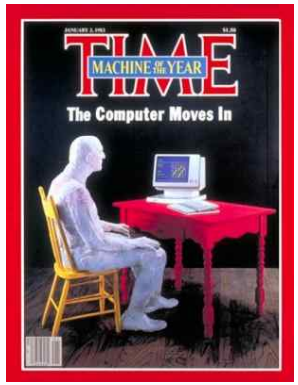
- Discrete Structure
- SW Dev. Fundamentals
- Algorithm & Complexity
- Architecture & Organization
- Operating Systems
- Networking & Communication
- Programming Languages
- Human-Computer Interaction
- Grophics & Visualization
- Intelligent Systems
- Information Management
- Social Issues & Professional Practice
- SW Engineering
- Computational Science
- Information Assurance & Security
- Parallel & Distributed Computing
- System Fundamentals
- Platform-based Development

Computer Science Body of Knowledge
(학사과정 학습내용)

[자료 출처 : http://www.acm.org/education/CS2013-final-report.pdf]

인공지능의 역사는 컴퓨터 발명이래 70년간의 신기술 부침의 역사

1983년 1월호



'50s-'60s
AI 개념 정립
다양한 방식 시도

'70
Expert System

'85
신경망

'00
자연언어처리

'10
딥러닝

'15
가상비서

?

1950
튜링테스트

1956
다트머쓰회의에서
AI 개념 정립

1970~80
MYCIN, XCON
Expert System

1986
Backpropagation
Multilayer Network

1997
IBM Deep Blue
체스 인간에게
승리

2000
Google Multiple
Language
Translatation

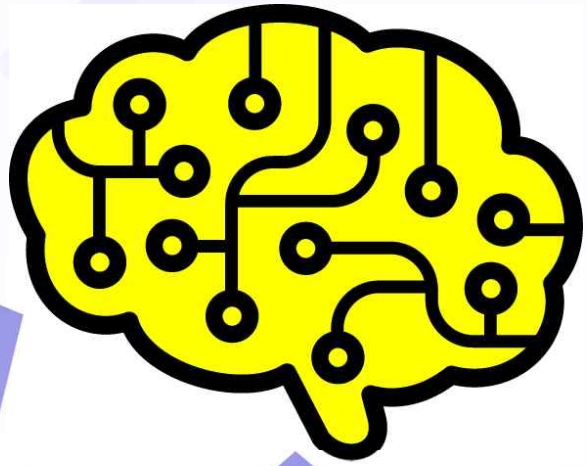
2011
IBM 왓슨
Jeopardy
승리

2012
Google
Deep Learning
고양이 인식

2014
Facebook
DeepFace

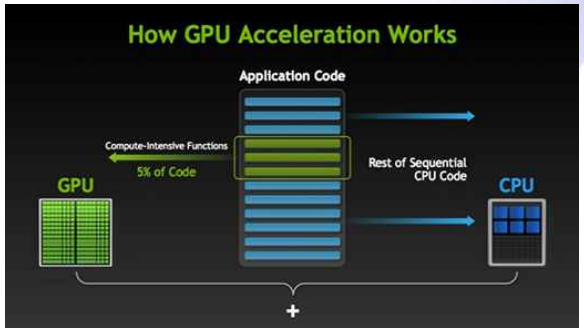
2016
AlphaGo
이세돌에 도전

인공지능 성공의 원동력



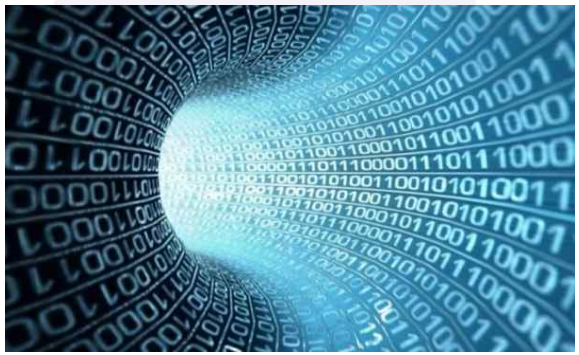
Computing Power

강력한 병렬 및 분산처리 능력



Big Data Power

인터넷, IOT, Sensor 기술을 통한 수집능력

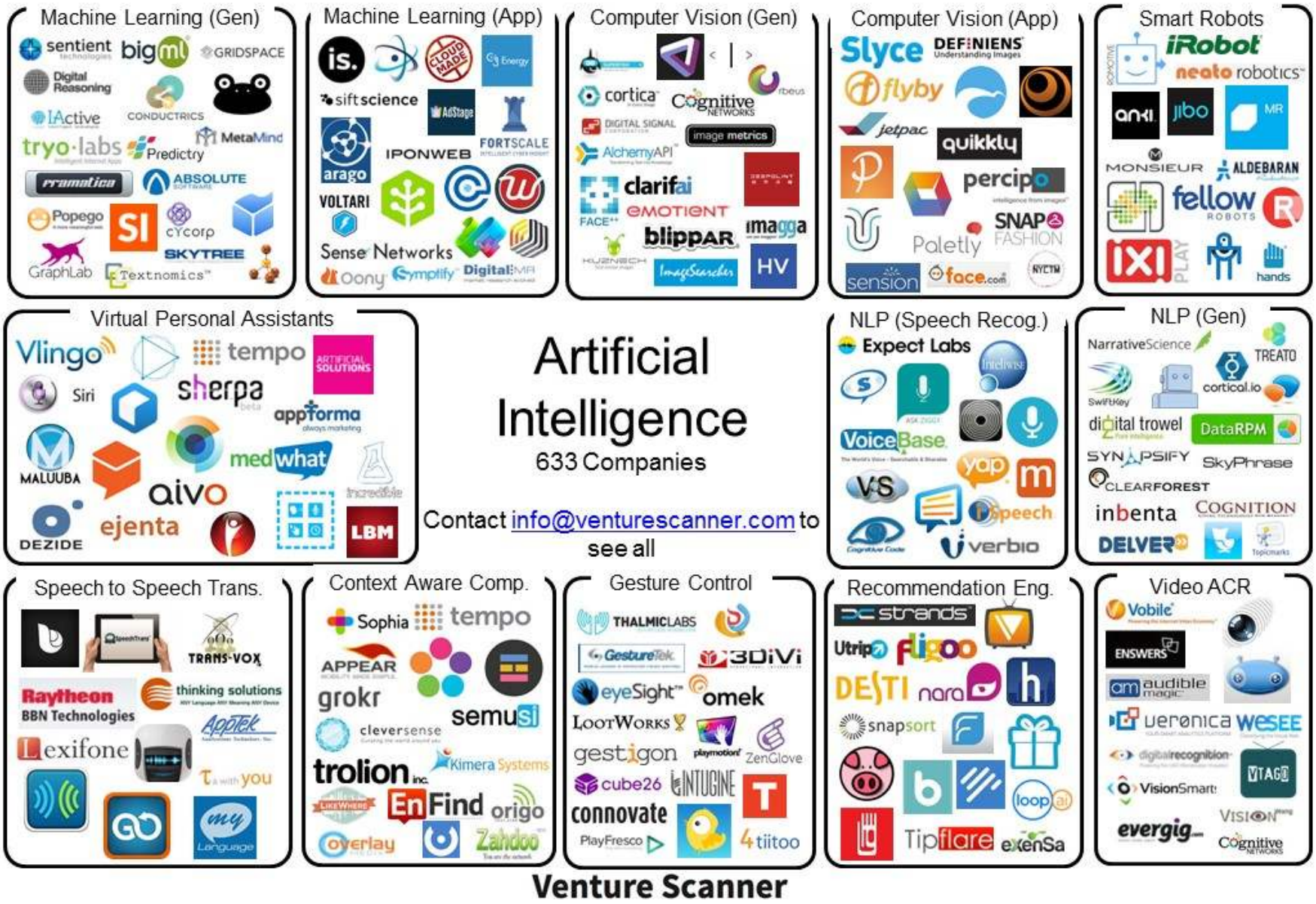


공개소프트웨어

개방·공유·협업의 성과

All companies are now AI company

:인공지능을 생산하거나 활용하는 기업



인공지능 개발 방법론

지식 처리형

- 사람의 지식을 기호의 조합으로 표현
- 이슈: 지식 획득 및 표현

데이터 기반형

- 신호데이터로부터 공통 성질을 추출
- 이슈: 훈련, 기계학습

전문가 시스템

IBM Watson

음성인식

영상인식

뉴럴 네트웍

딥러닝

가상 비서

그림 내용 설명하는 로봇

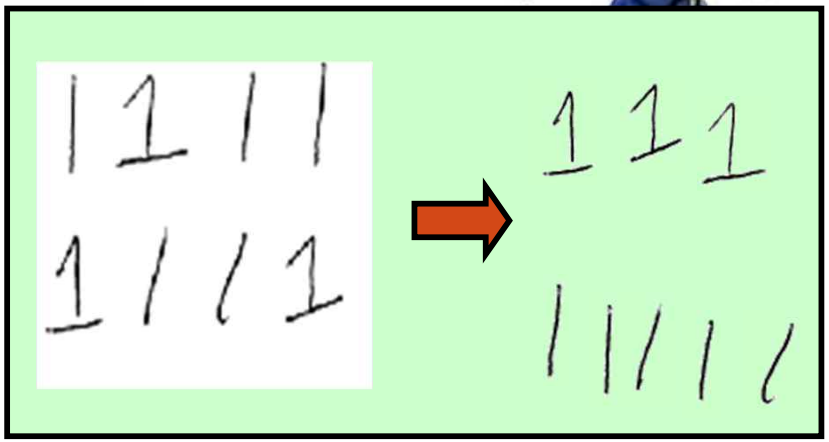
기계학습(Machine Learning)

**명시적으로 프로그램하지 않고,
스스로 학습할 수 있는 능력을
컴퓨터에게 주기 위한 연구**

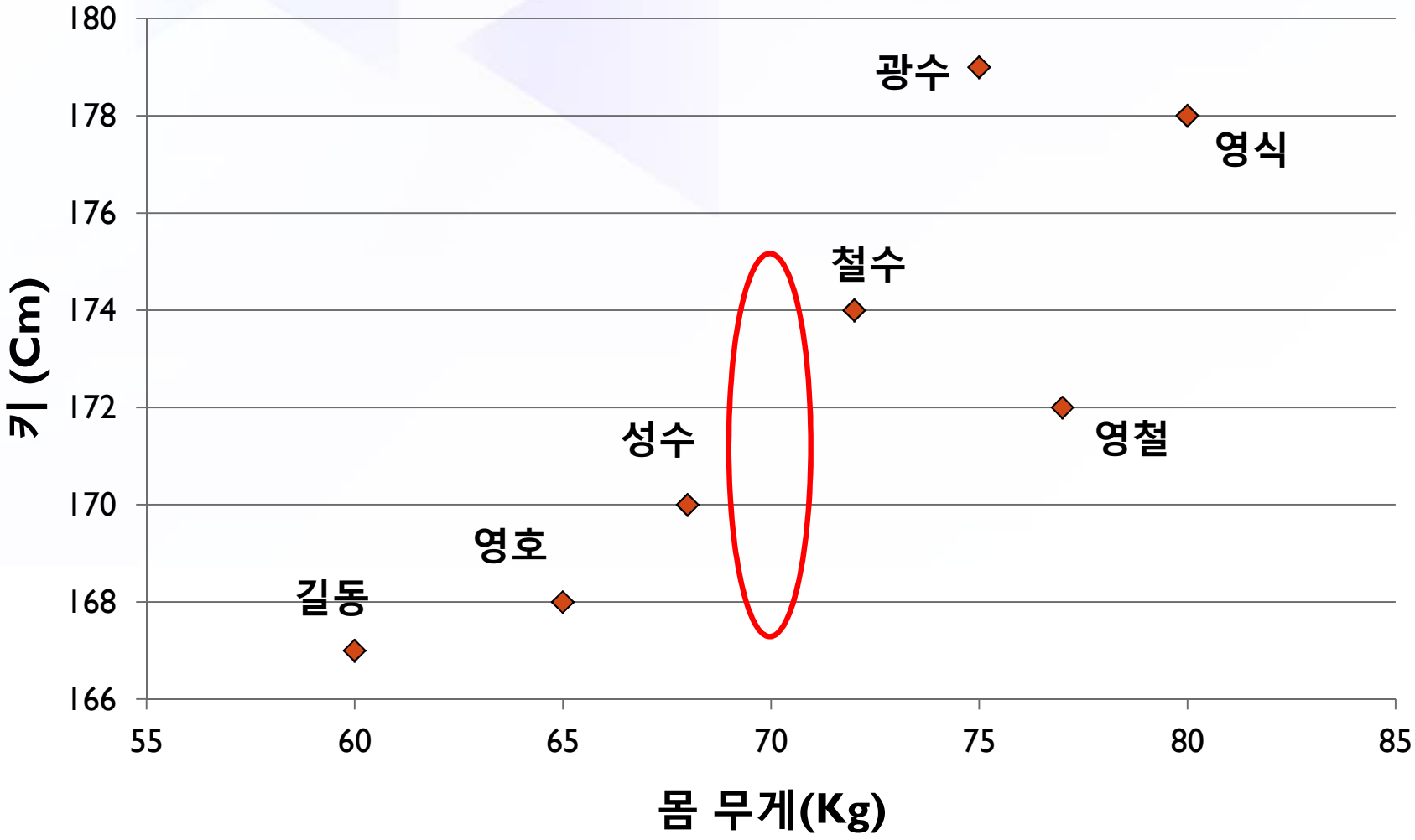
소프트웨어 중심사회의 Think Tank SPRi Software Policy & Research Institute

학습 알고리즘의 종류

- **지도학습(Supervised learning)**
  - 올바른 입력/출력 쌍으로된 훈련 데이터로부터 입출력간의 함수 학습
- **자율학습(Unsupervised learning)**
  - 데이터의 무리 짓기(Clustering) or 일관된 해석의 도출
- **증강 학습(Reinforcement learning)**
  - 계속된 행동으로 얻은 보상으로부터 올바른 행동을 학습
- **…**

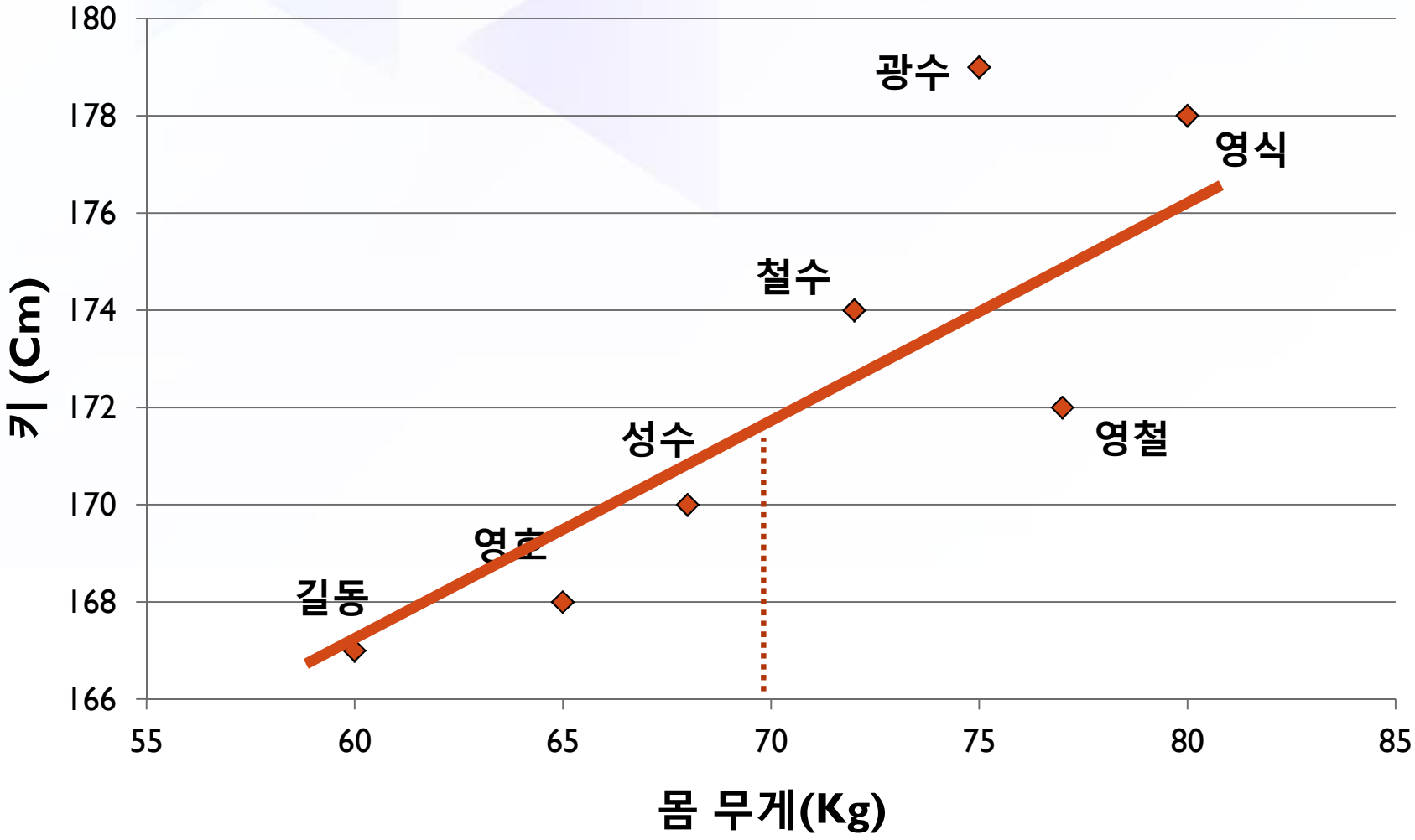


소대원의 (몸무게, 키) 데이타



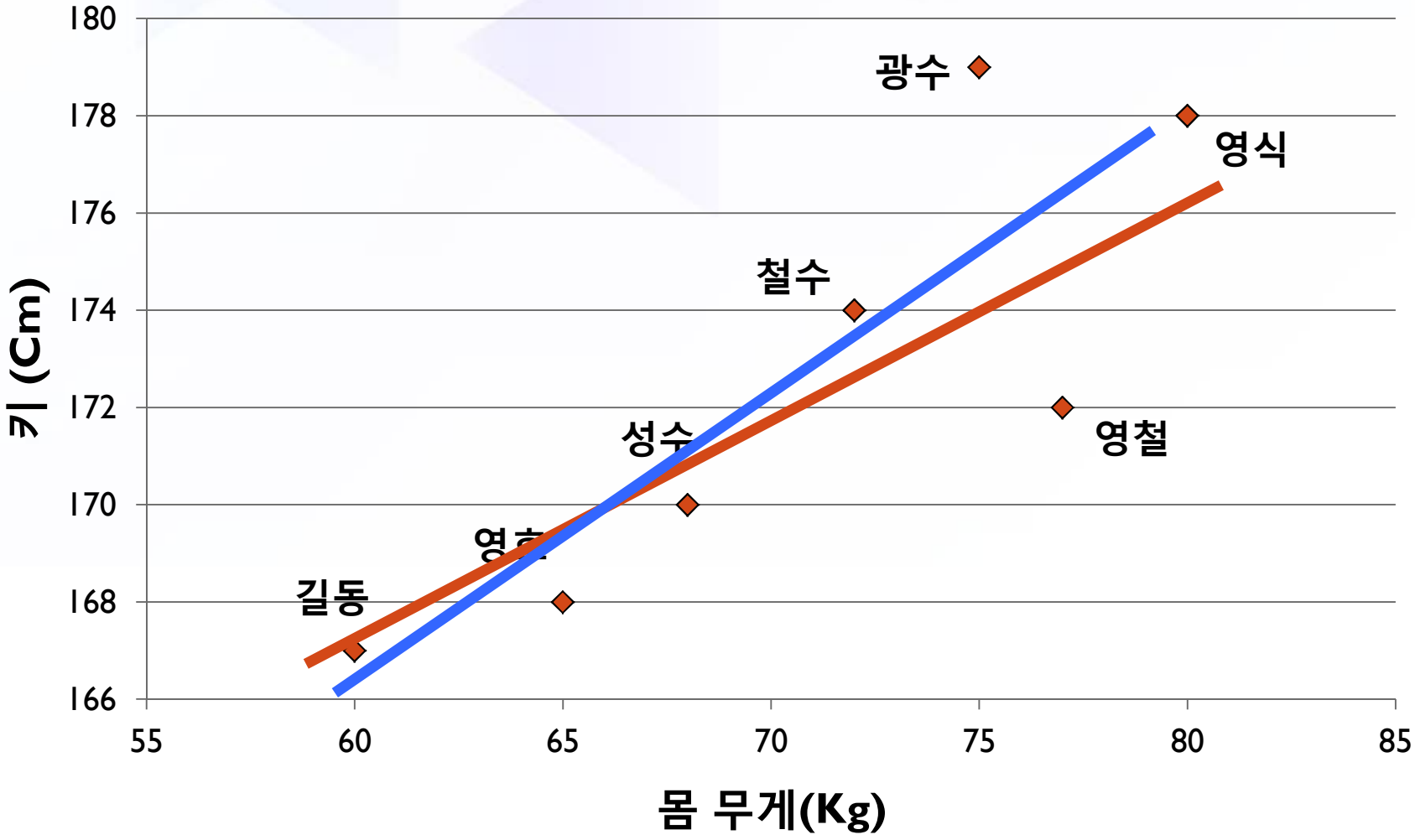
몸무게 70Kg인 '개똥이'의 키를 예측해 보자

몸무게와 키의 관계를 선형 함수로



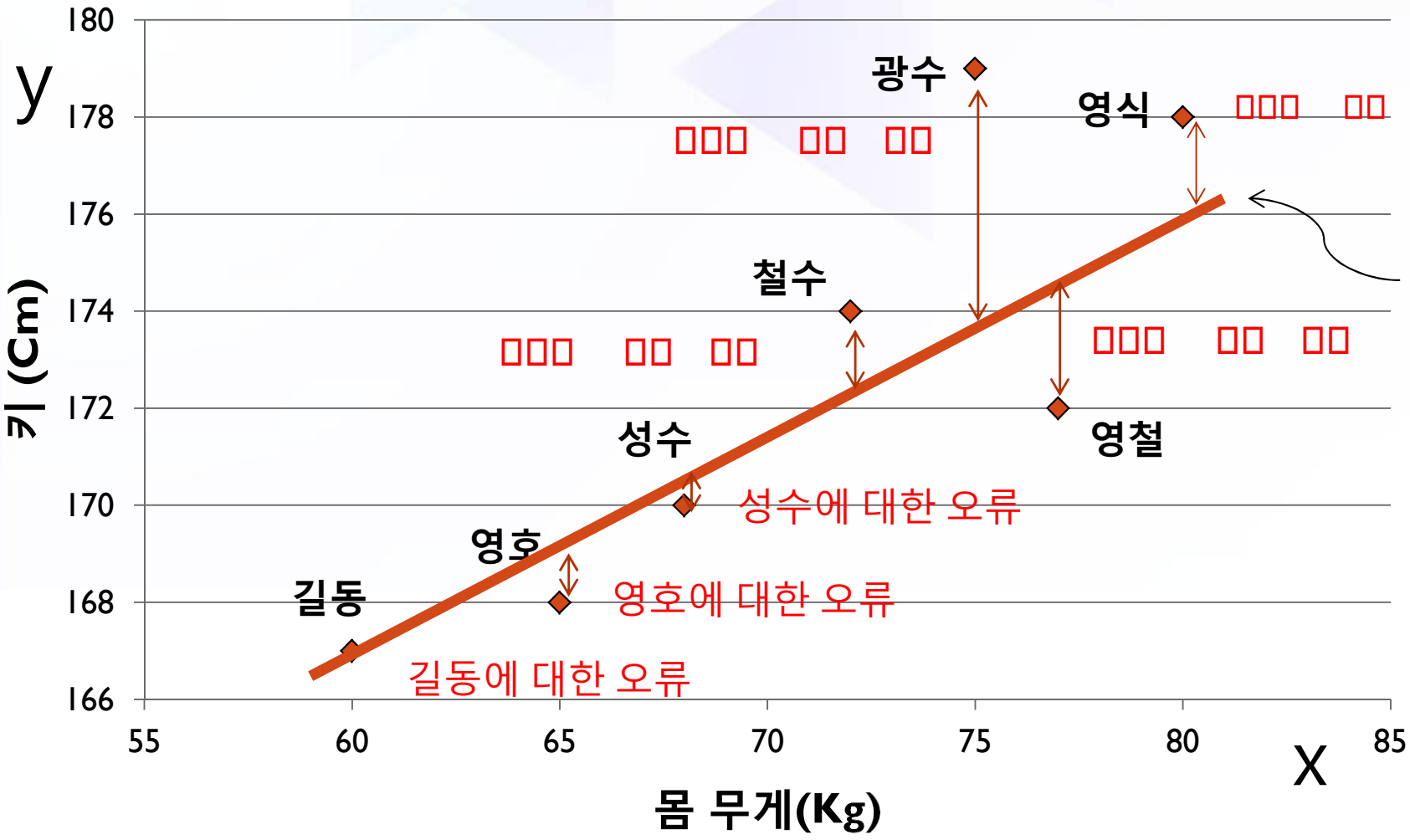
몸무게 70Kg인 개똥이의 키는 ?

직선 중에서 가장 좋은 것은?



무수히 많은 직선 중에서 가장 좋은 것 찾기

오차의 합이 가장 적은 직선 찾기



직선의 식

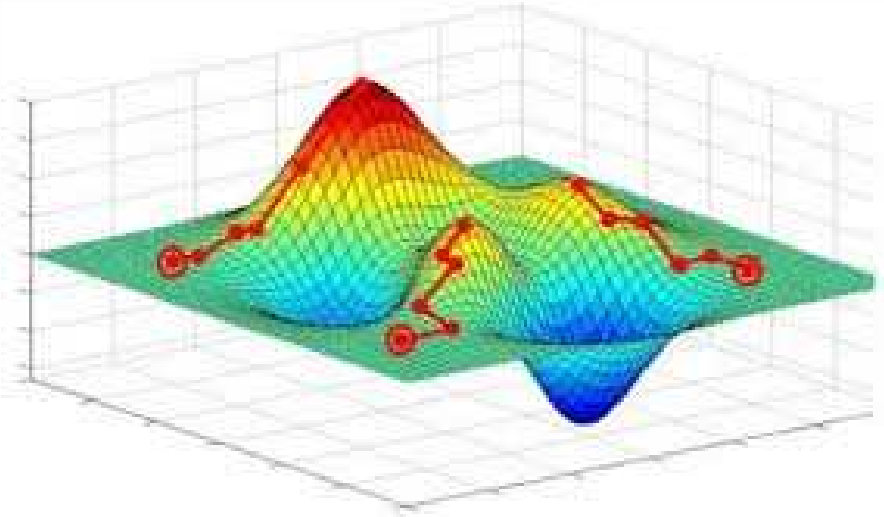
$h(X) = aX + b$

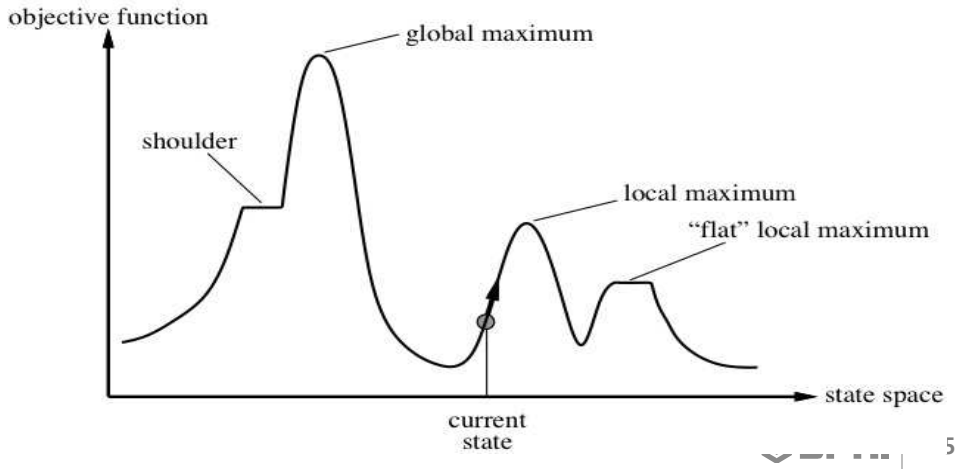
a, b를 변화시켜가면서 J가 최소되는 a와 b를 찾음

$$J = \frac{1}{2}\sum_{i=1}^{m}(h(xi) - yi)^2$$

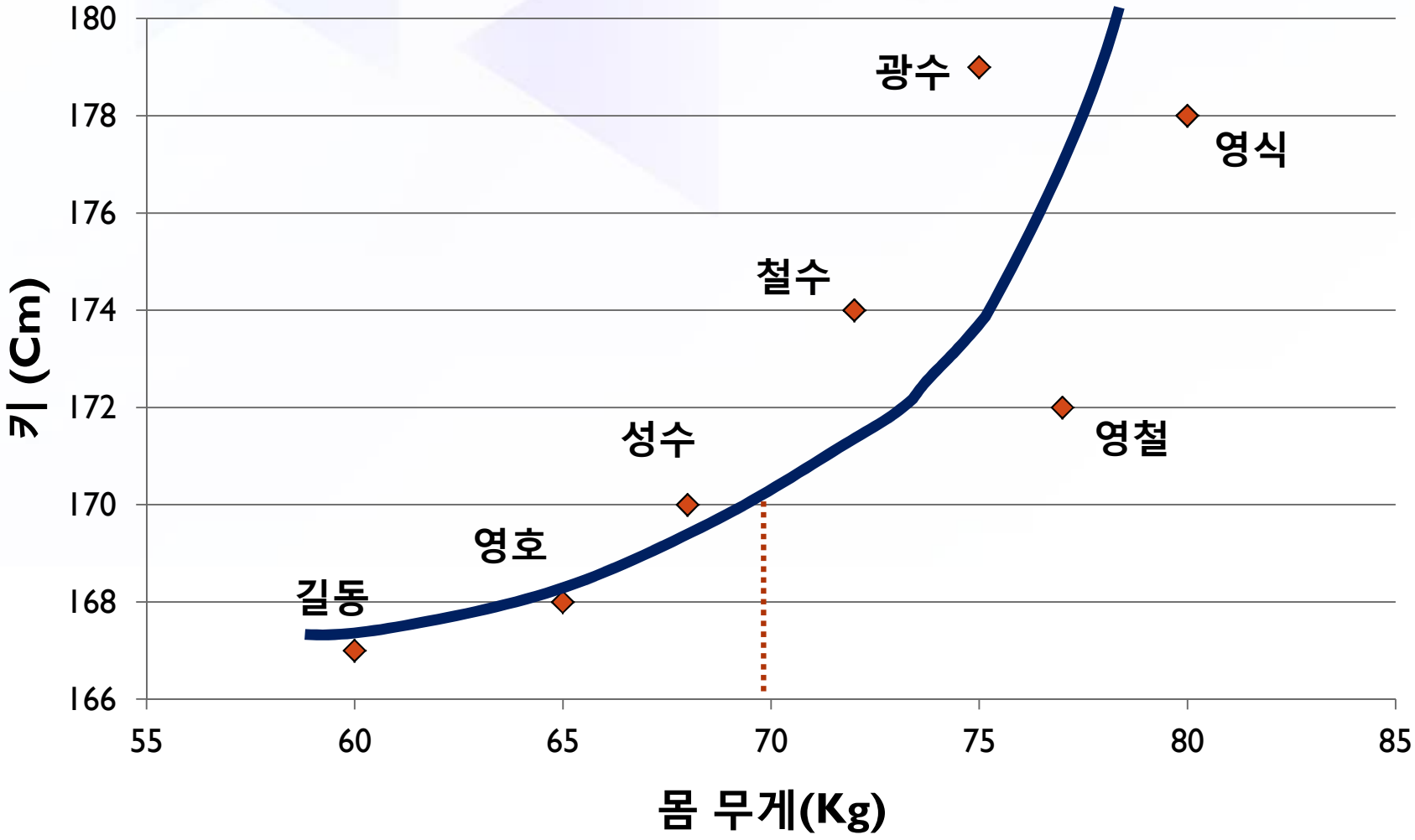
함수의 최대(최소), 즉 극점 위치 찾기

- **Gradient Descent (급한 기울기 따라가기)**
  - 반복하여 가장 경사가 급한 곳으로 Parameter를 변화시켜 최대(최소)점 도달
- **복잡한 함수의 최적화에 많이 사용, 특히 신경망 학습 등에서**
- **문제점**
  - 시작 위치에 따라서 종종 Local 극점에 도달
  - 특이 지형에서 방향 상실
  - 얼마만한 보폭으로?



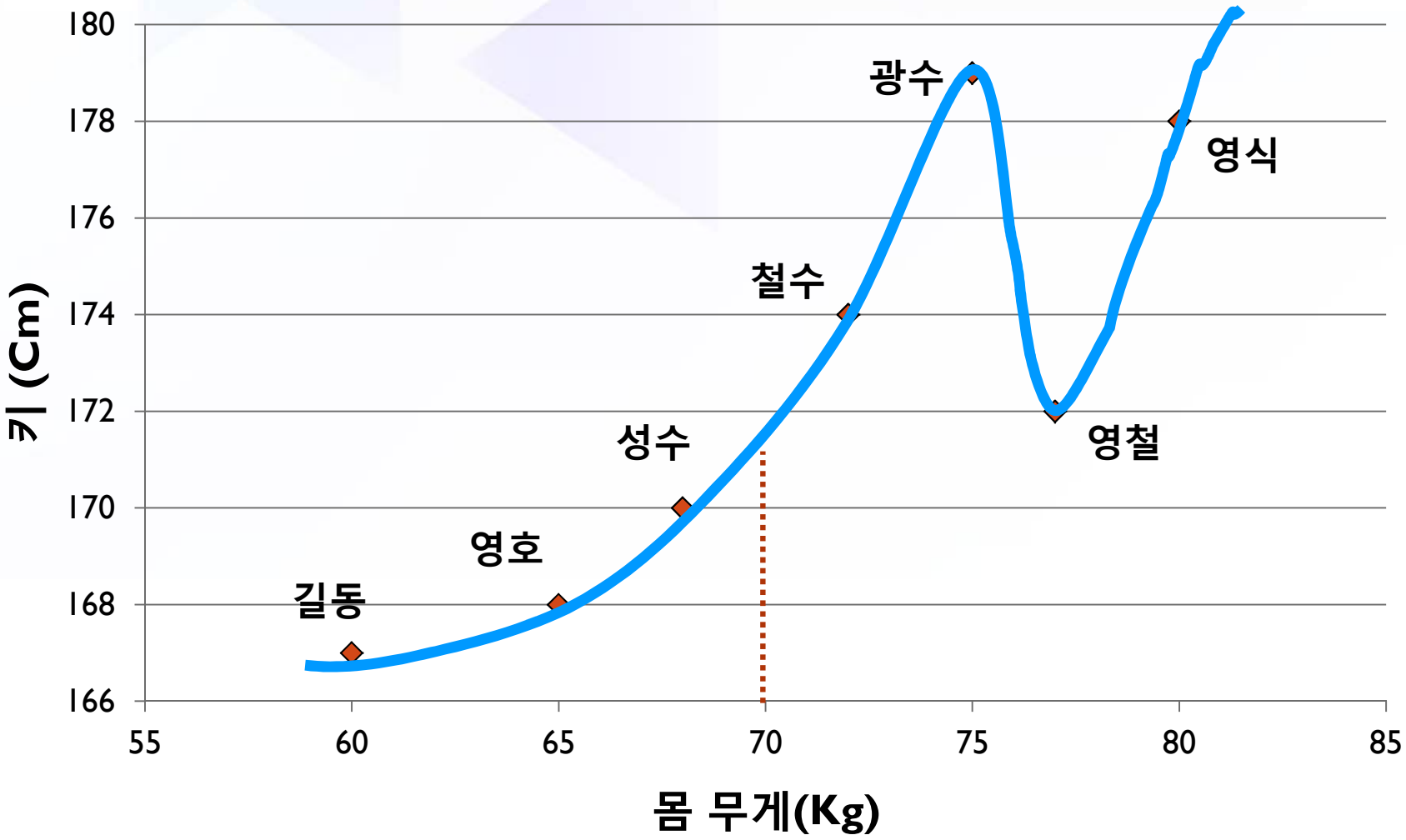


몸무게와 키의 관계를 2차 함수로



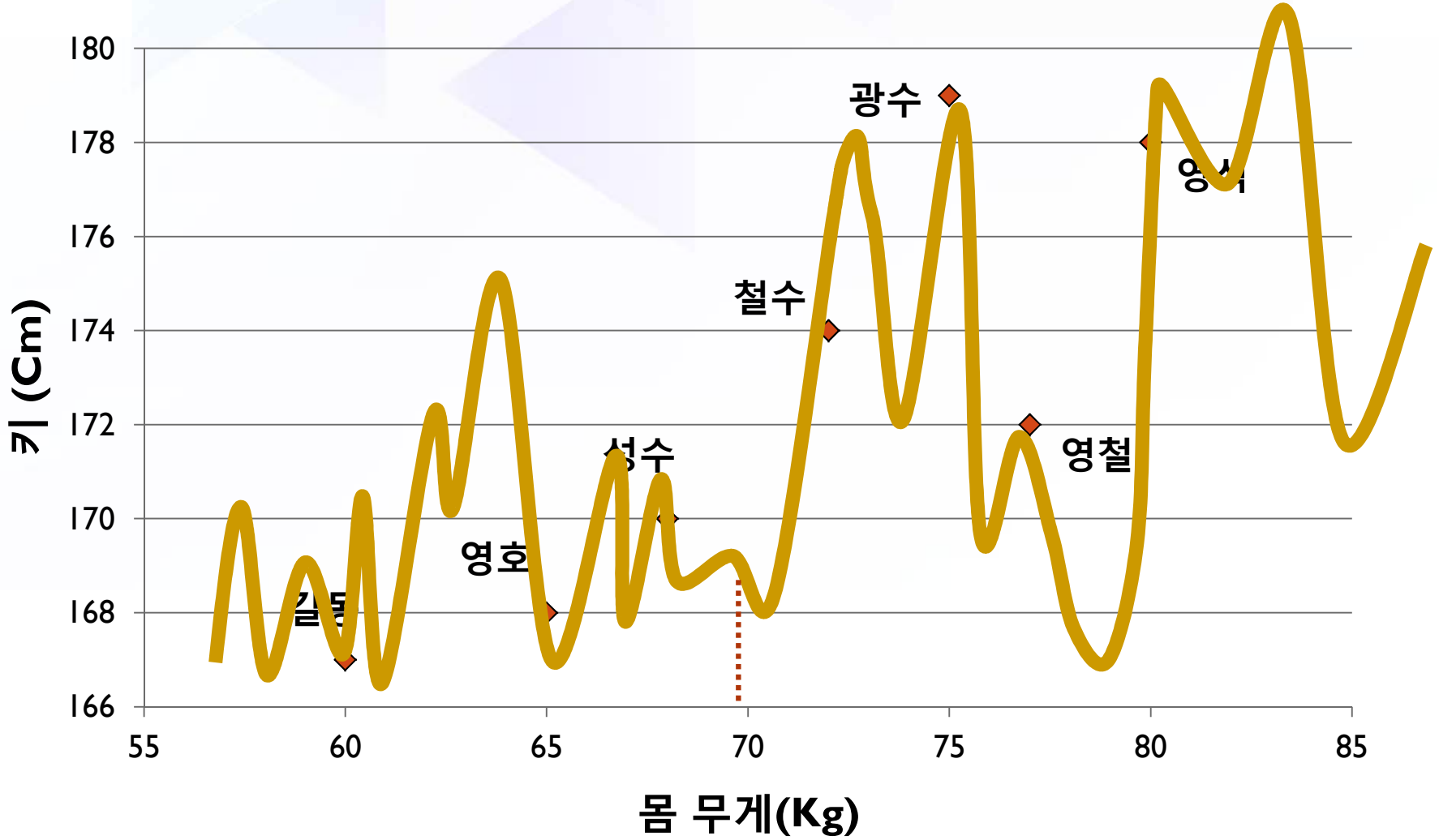
1차 함수의 경우보다 오류의 합을 줄일 수 있다

몸무게와 키의 관계를 3차 함수로



모든 데이터에서의 오류를 없앨 수 있다

몸무게와 키의 관계를 복잡한 선형 함수로



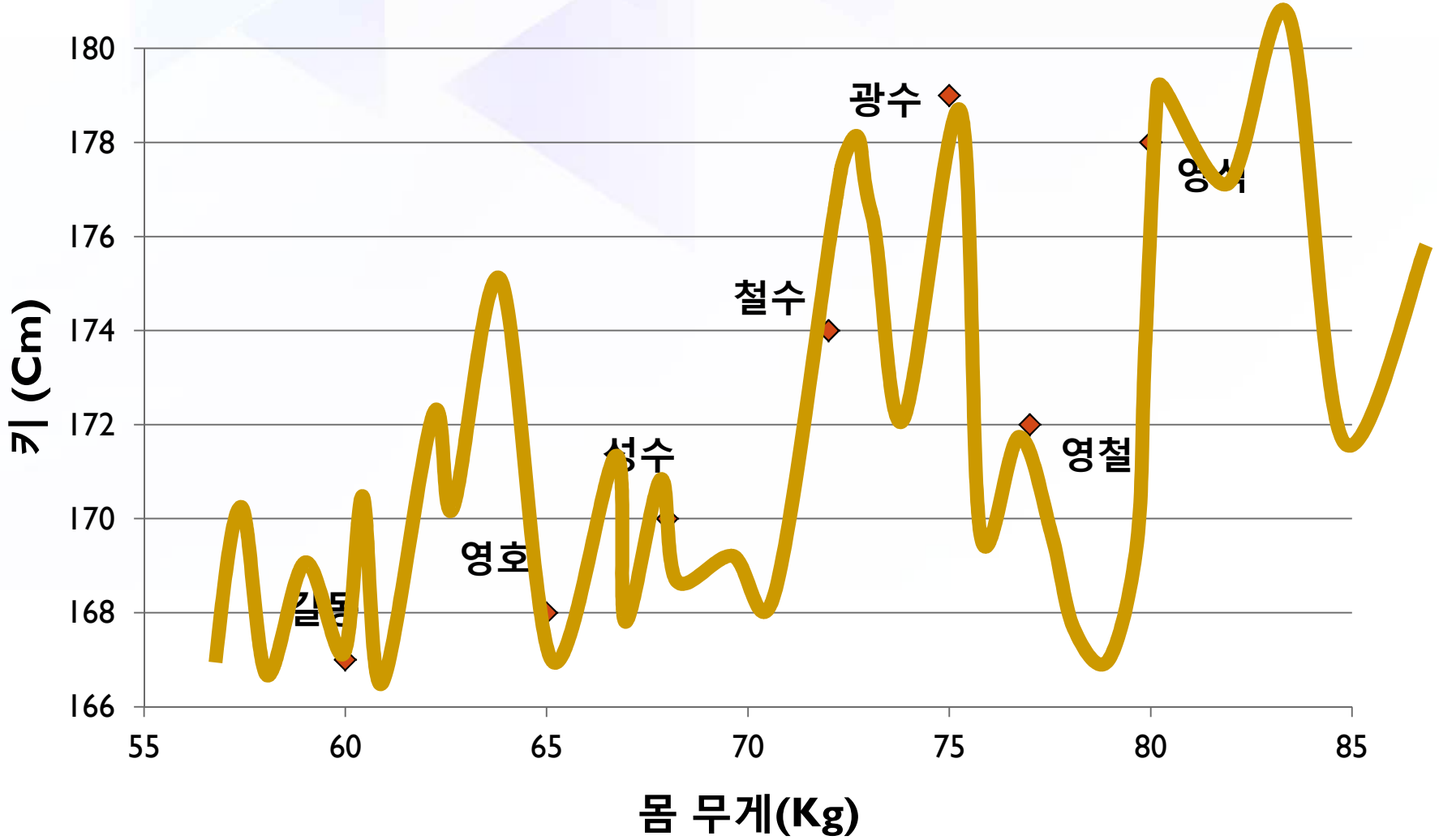
모든 데이터에서의 오류를 없앨 수 있다

1차, 2차, 3차, N차 함수 중에서 어느 것이 가장 좋은가?

어떤 철학에 의하여 "좋음"을 정의

- 예) Ockham's Razer
  - 단순한(Simple) 것을 선호
- 예) 함수의 복잡도와 데이터 적합성 간의 타협(Tradeoff)
- 예) 일반화 능력이 강한 것
  - 훈련에 참여하지 않은 데이터에 대하여도 좋은 성능을 보일 것

높은 차원의 함수로 관계를 표현하는 것은



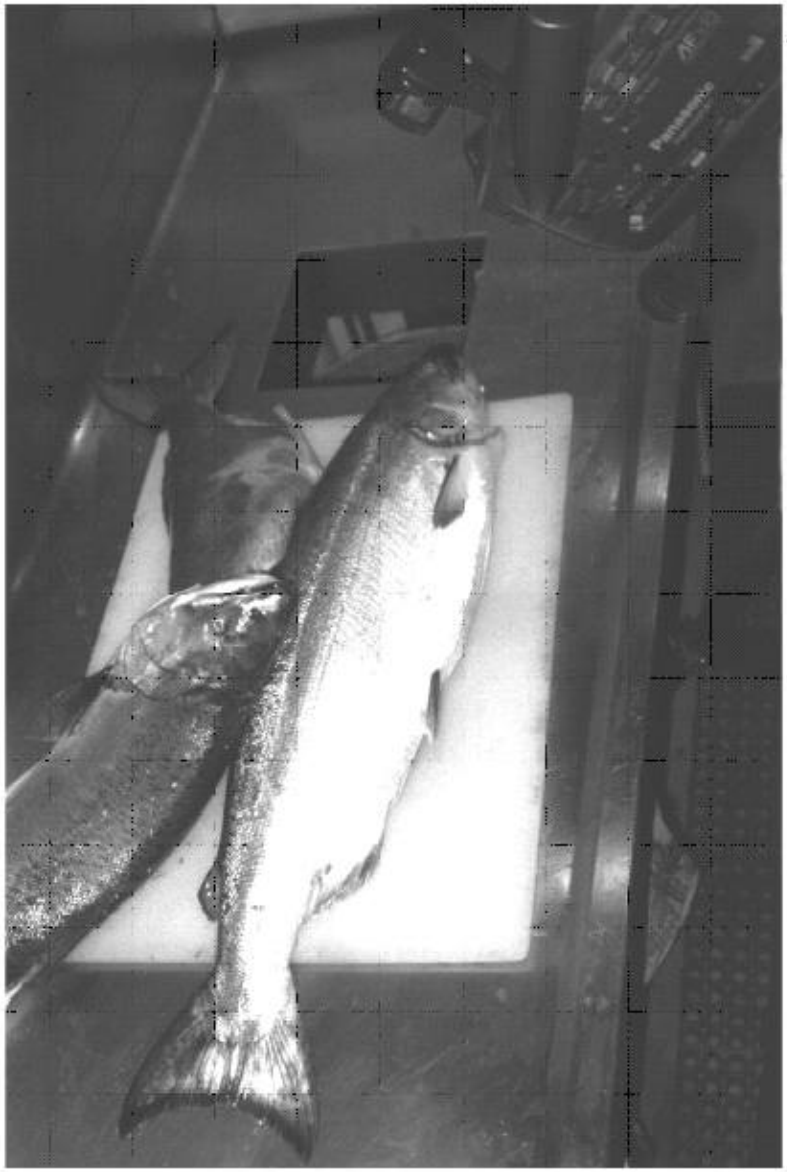
일반화 능력에서 바람직하지 않다

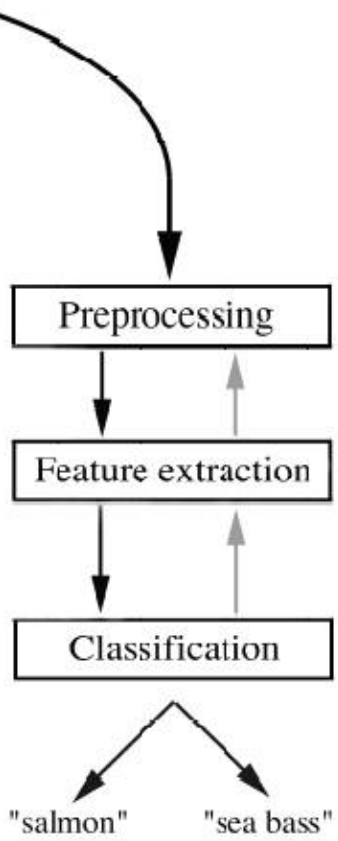
패턴인식 시스템 설계

패턴인식은 지능이 필요한
일반적인 문제풀이 능력

소프트웨어 중심사회의 Think Tank SPRi Software Policy & Research Institute

연어-농어 분류 문제





특징을 추출

그 특징을 보고 분류

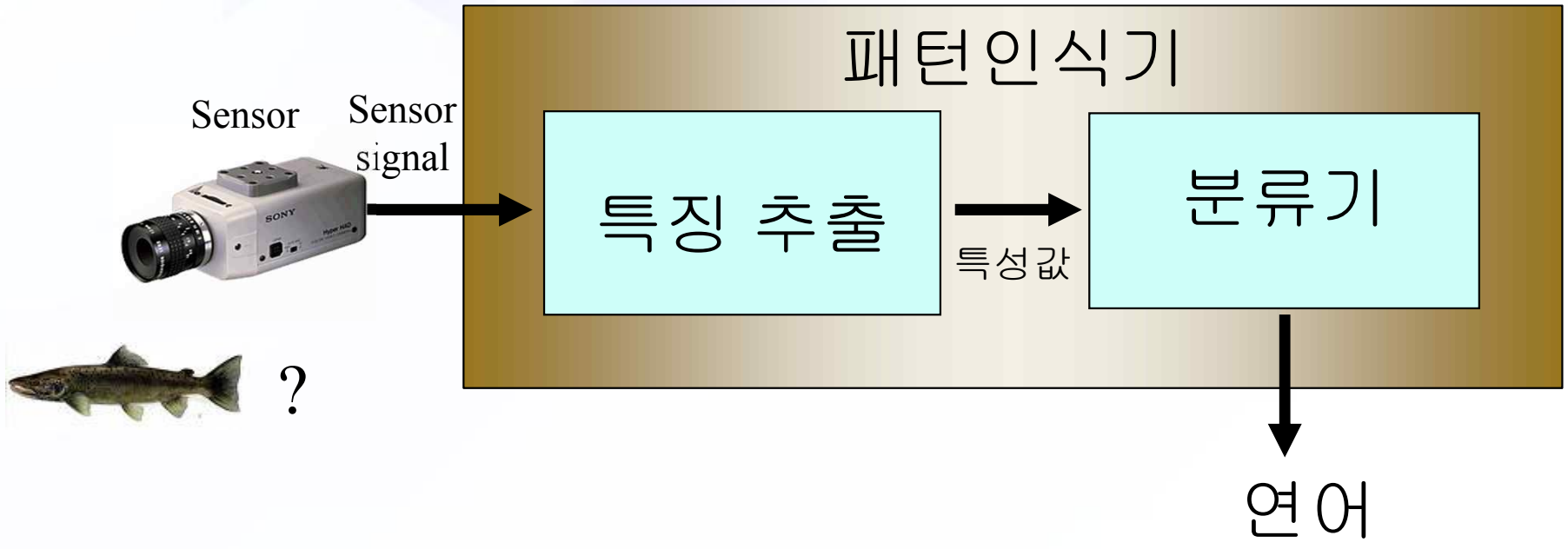
농어(Sea Bass) vs 연어(Salmon)

농어

연어

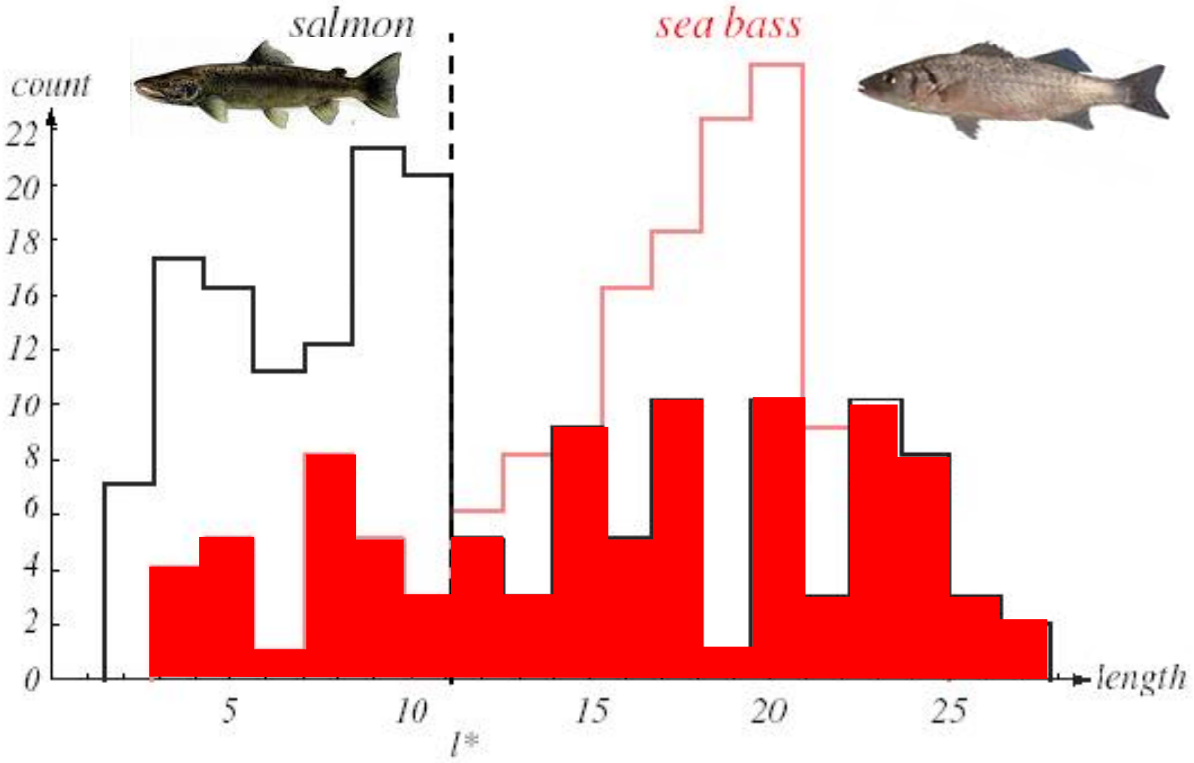
Length
Lightness
Number and shape of fins
Position of the mouth

패턴인식과 기계학습



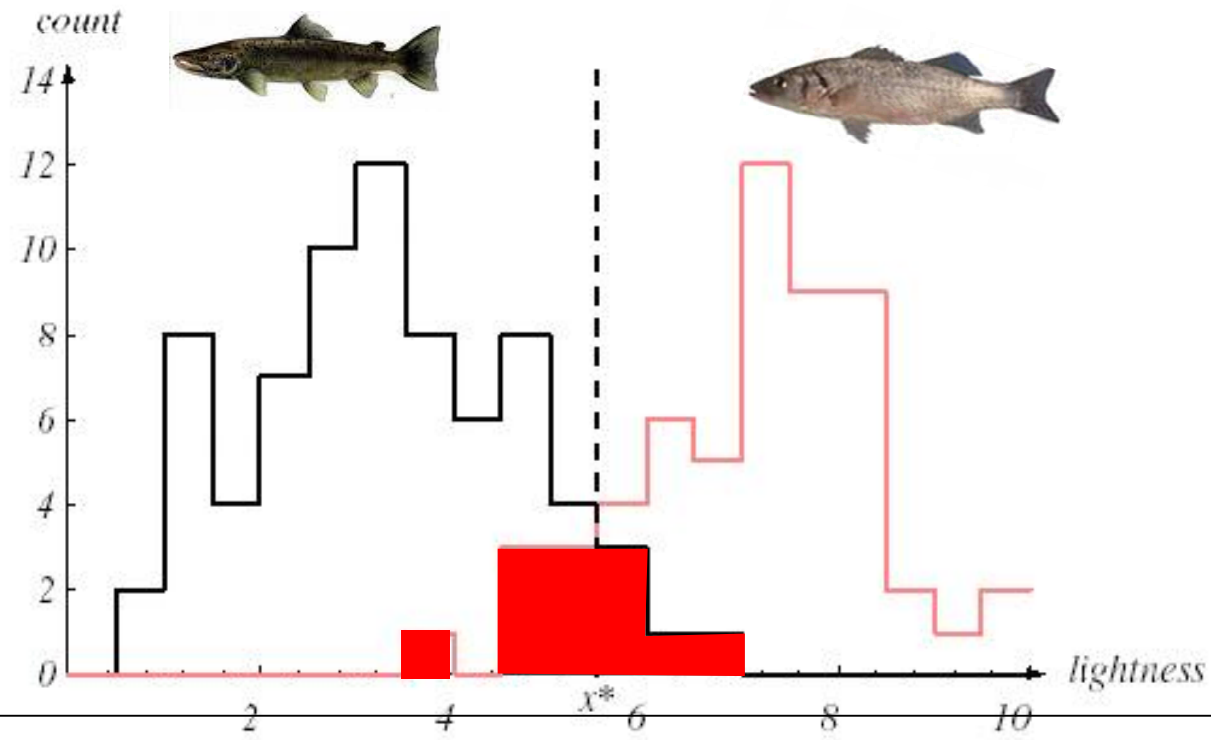
- 어떤 특징을 보고 어떻게 분류 할 것인가를 결정하는 것이 기계학습의 역할
- 패턴인식과 기계학습은 동전의 앞 뒷면!

‘길이’특성을 보고 분류하겠다면



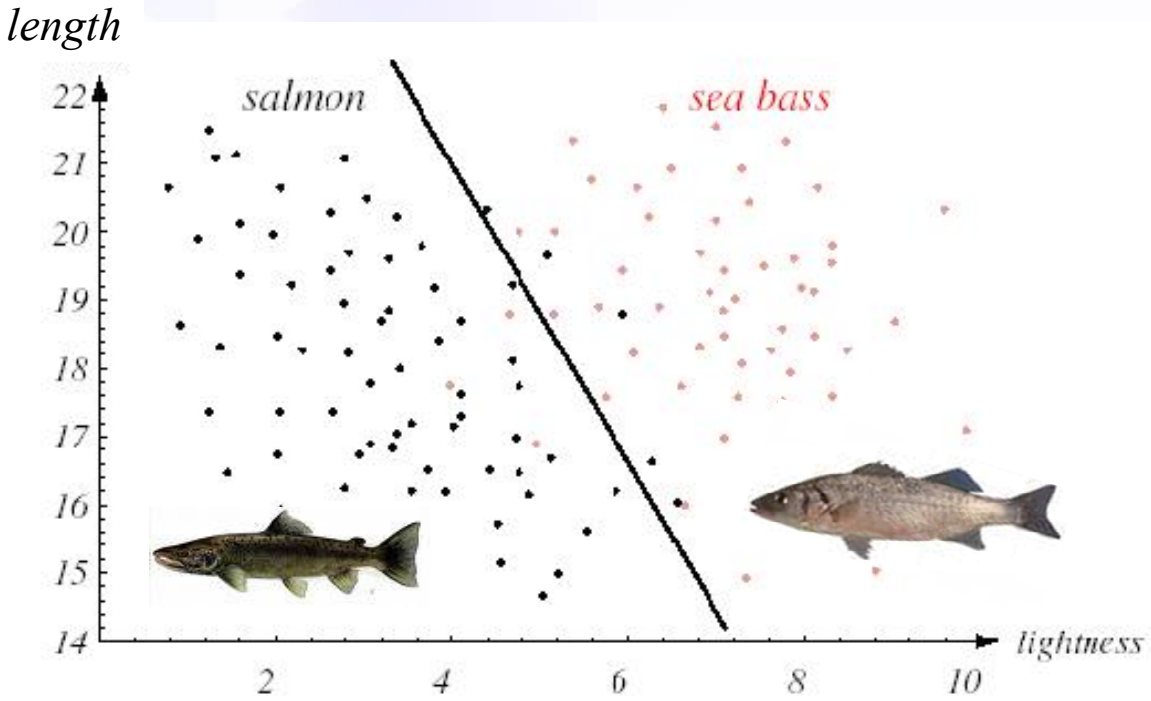
“길이가 l보다 크면 연어라고하고 작으면 농어라하자”
오류를 최소화하는 l^* 구할 수 있다

‘밝기’ 특성을 보고 분류하겠다면



- “밝기가 x보다 어두우면 연어라고 하고 밝으면 농어라 하자”
  - *“길이를 보고 분류하면 밝기에 의한 분류보다 오류가 많다”*
  - 오류를 최소화하는 x* 구할 수 있다
- 특성의 선택이 패턴인식의 성패를 좌우
- 특성이 주어지면 훈련데이터로부터 최적의 분류 방법 학습가능

두 특징을 같이 보면?

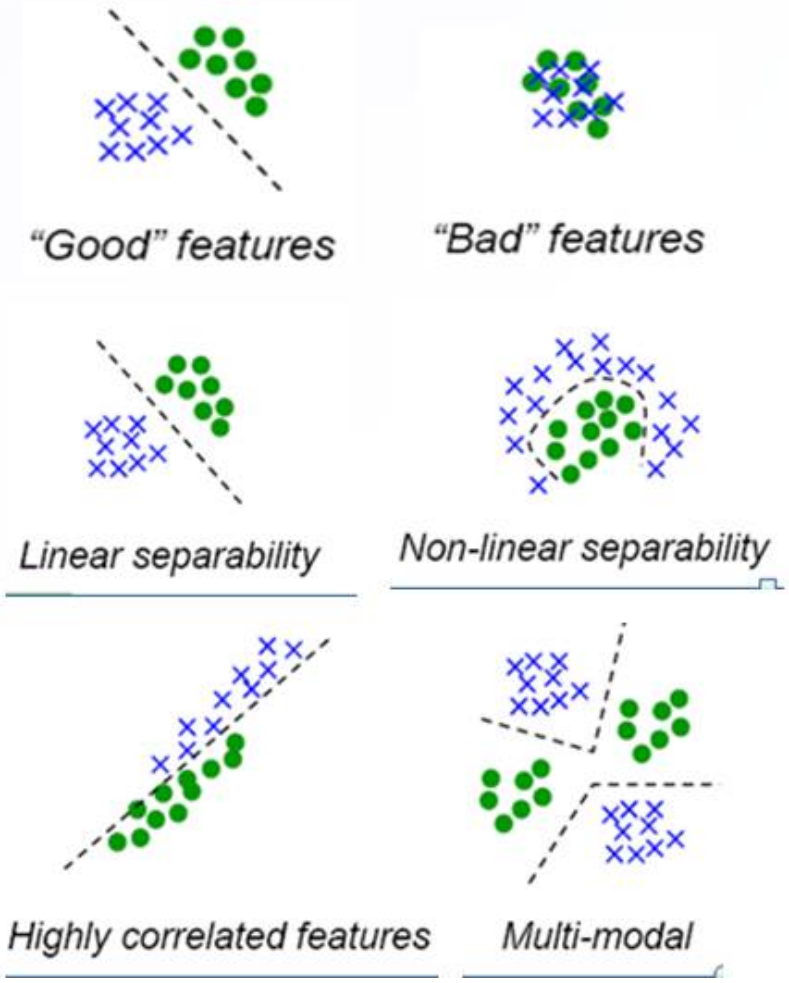


더 많은 특징을 보면 더 좋을까?

길이 밝기, 폭, 핀의 개수, 핀의 모양, 입의 위치, ...

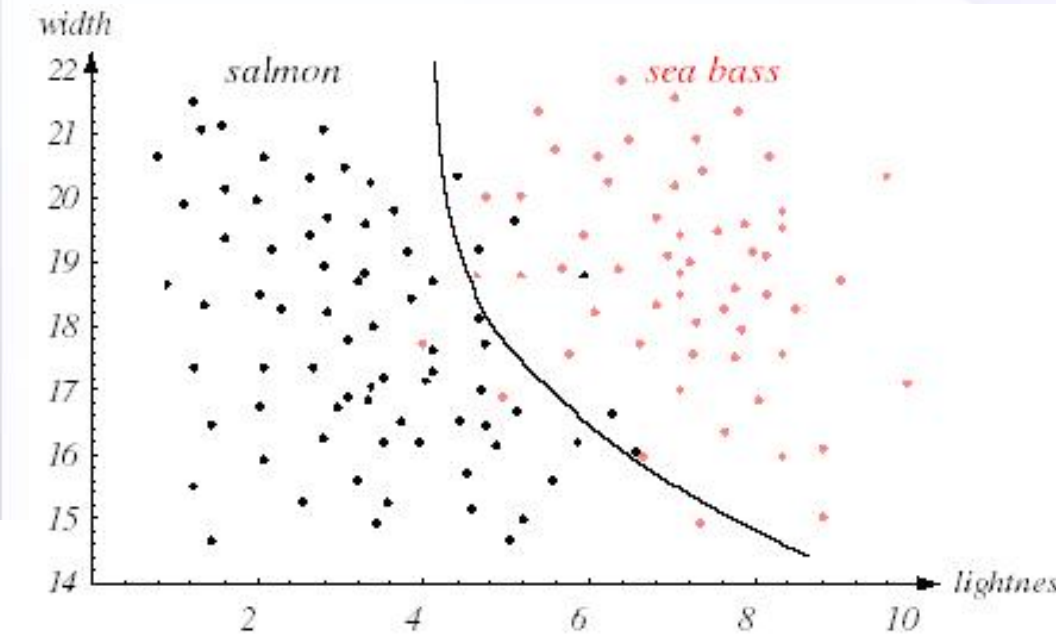
특성과 패턴인식의 성능, 학습의 용이성

- 특성의 선택이 패턴 인식 성능 좌우
- 더 많은 특성은 더 좋은 성능?
- 상관 관계가 깊은 특성은 성능을 향상시키지 못한다
- 특성에 따라 학습의 어려움이 상이

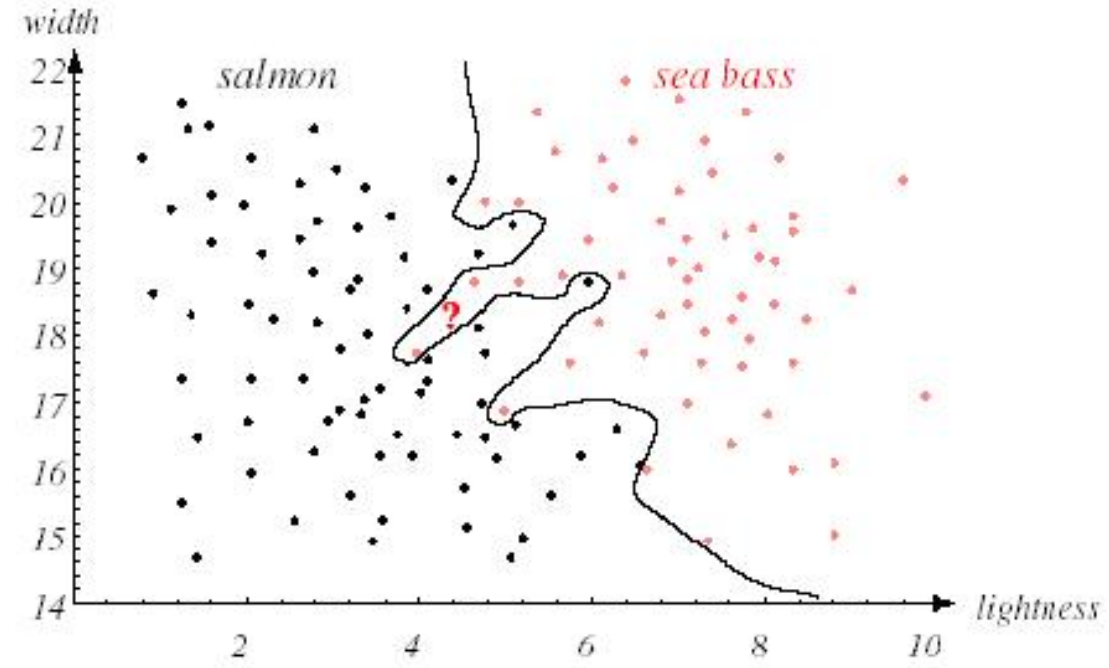


일반화 능력

훈련에 참여하지 않은 데이터에 얼마나 좋은 성능을 보일까?



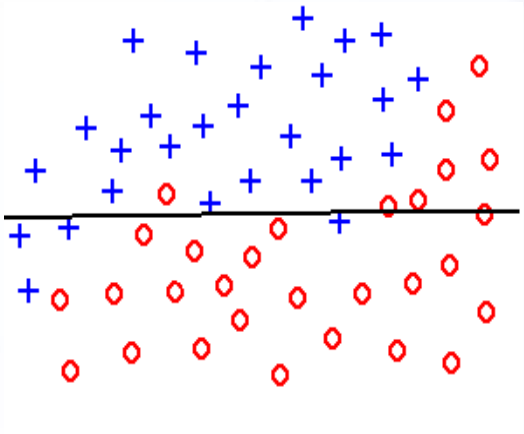
Simple Model A



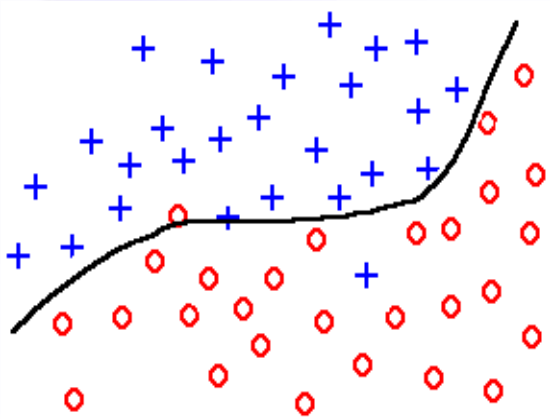
Complex Model B

- 훈련 데이터에는 완벽하도록 복잡한 모델을 고를 수 있다.
- 일반화 능력이 최상이 되도록 모델을 선택해야

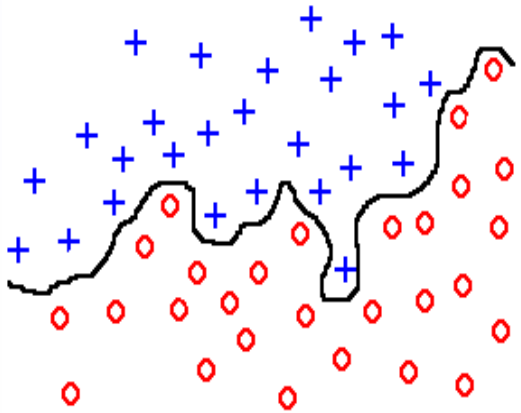
Overfitting and Underfitting



underfitting



good fit

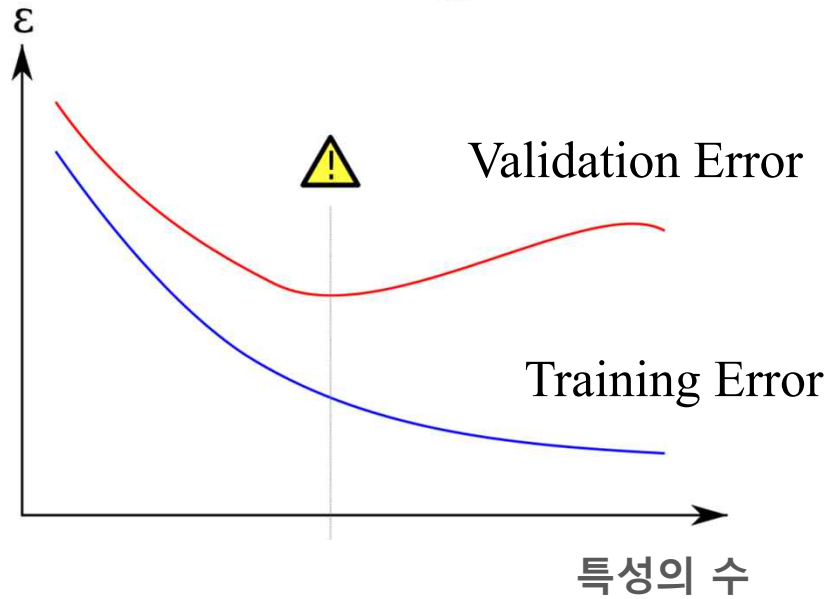


overfitting

- 복잡도에 대한 패널티를 포함하여 최적화

$$\min_f \sum_{i=1}^{n} V(f(\hat{x}_i), \hat{y}_i) + \lambda R(f)$$

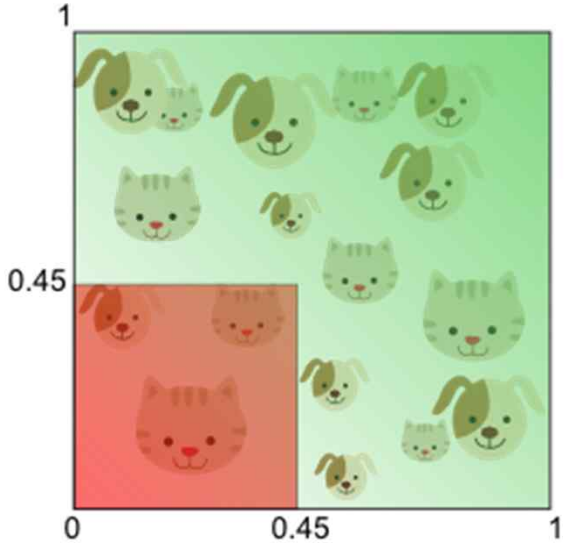
- 훈련데이터와 별도의 검증데이터로 최적의 모델 선택

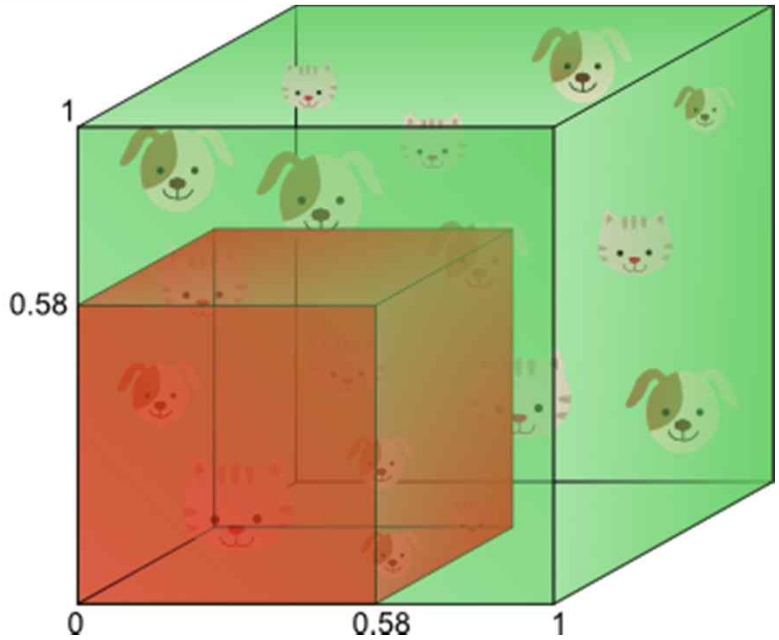


특징의 추가할수록 차원이 증가

- 유한한 데이터를 가지고 특징을 추가하면 공간의 크기가 너무 급하게 증가하여 가용한 데이터가 희소화
  - 통계적으로 신뢰할 수 있는 결과 얻기 어려움
- 적절한 일반화 능력을 위해 요구되는 훈련데이터 양은 급격히 증가하는 현상을 "**차원의 저주 (Curse of Dimensionality)**"라함

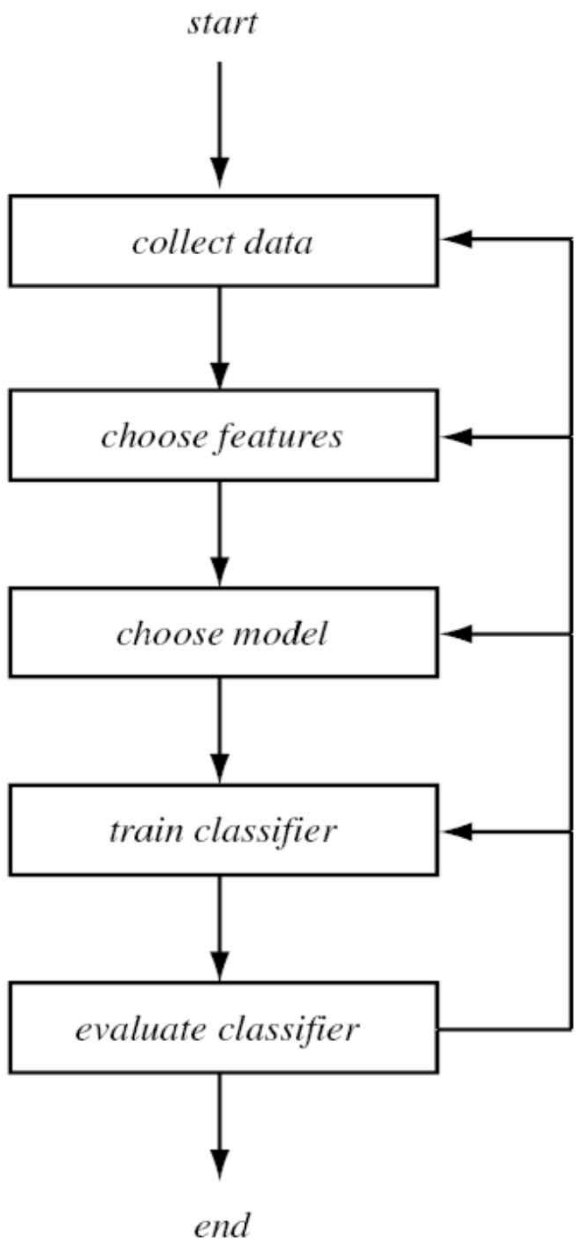
예: 총 Sample 수의 20%를 확보하기 위한 차원별 노력



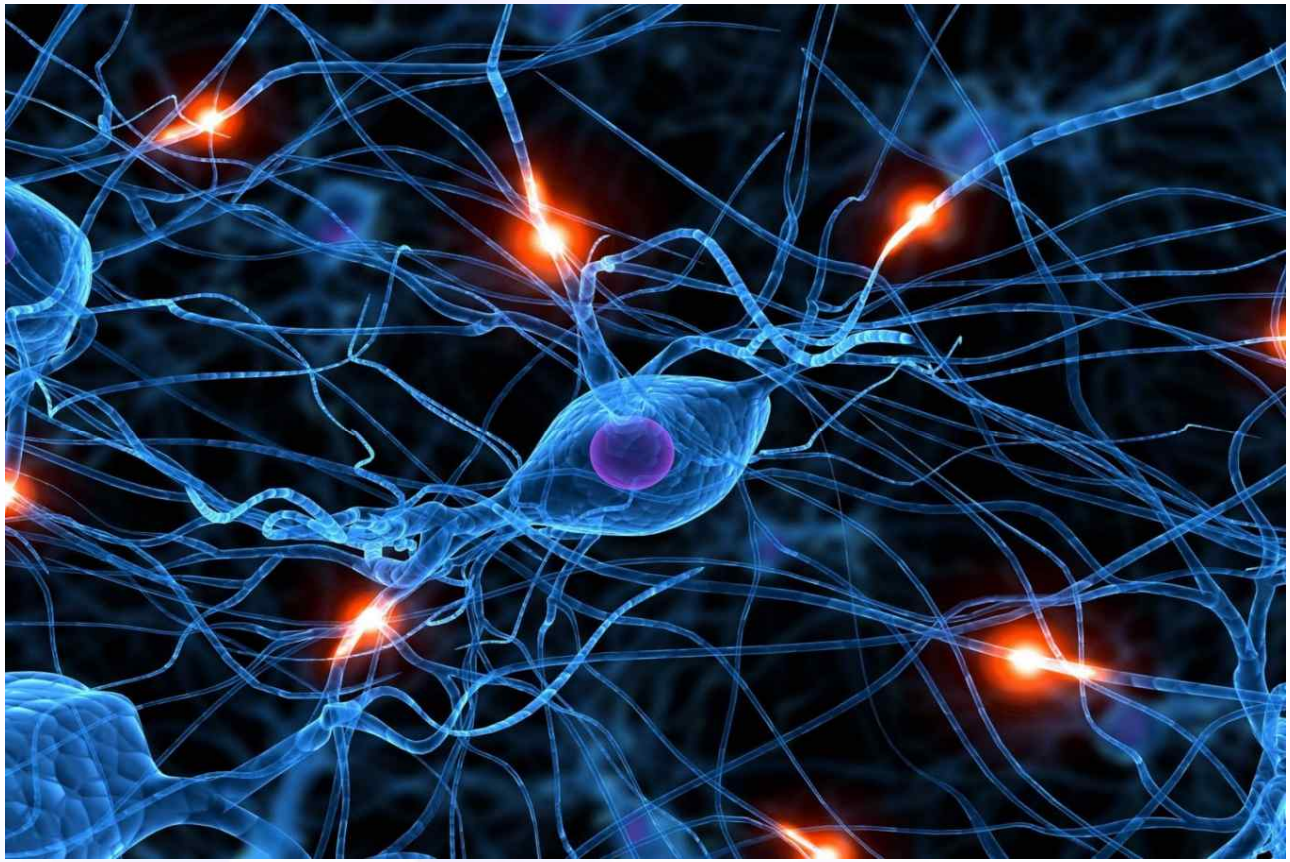


고전적 패턴인식 시스템의 설계

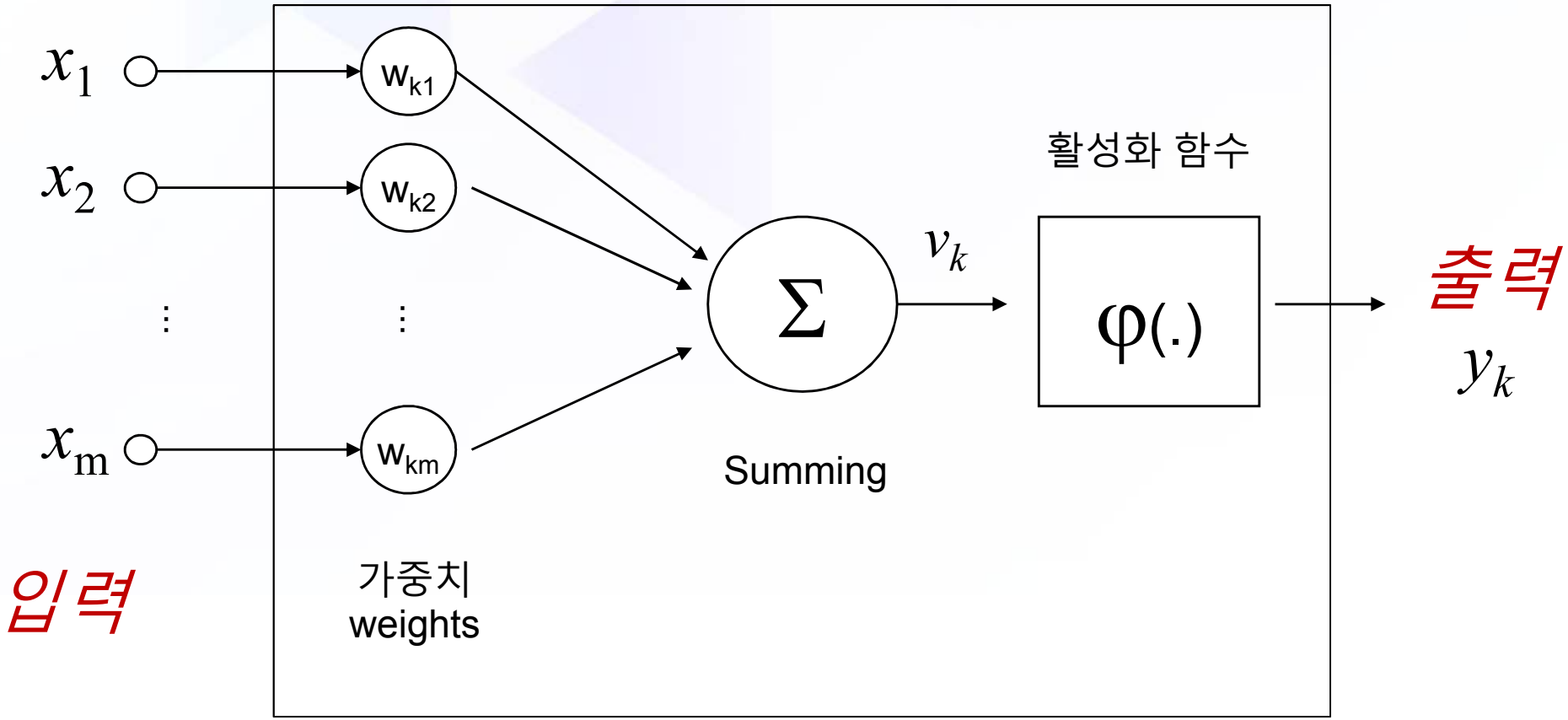
- **데이터 수집**
  - Probably the most time-intensive component of project
  - How many examples are enough ?
- **특성의 선택**
  - Critical to the success of the PR project
  - Require basic prior knowledge, engineering sense
- **인식 방법론의 선택과 설계**
  - Statistical, neural and structural
  - Parameter settings
- **훈련**
  - Given a feature set and 'blank' model, adapt the model to explain the training data
  - Supervised, unsupervised, reinforcement learning
- **평가**
  - How well does the trained model do ?
  - Overfitting vs. generalization



신경망(Neural Network)

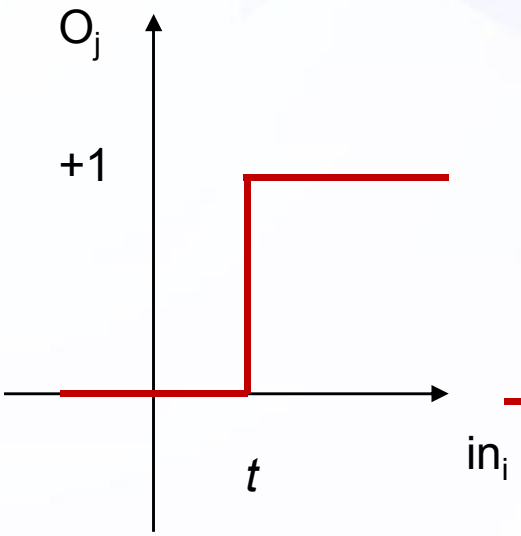


신경 세포의 수학적 모형

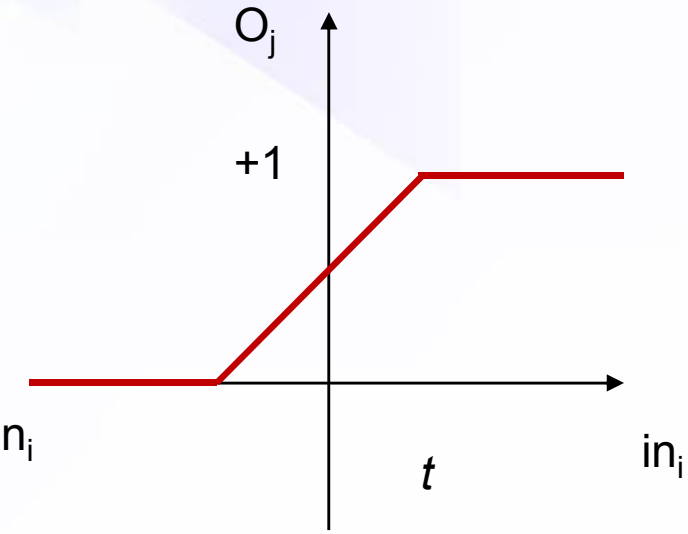


$$v_k = \sum_{j=1}^{m} w_{kj} x_j \qquad y_k = \varphi(v_k)$$

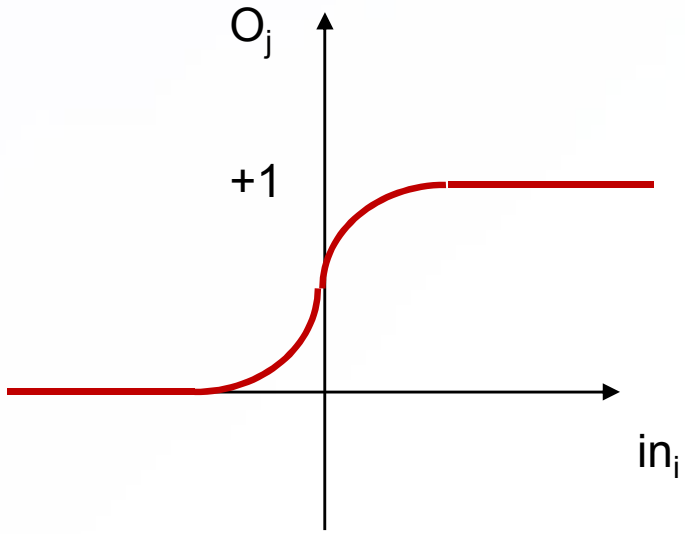
활성화 함수의 형태 $\varphi(v)$



Threshold Function



Piecewise-linear Function

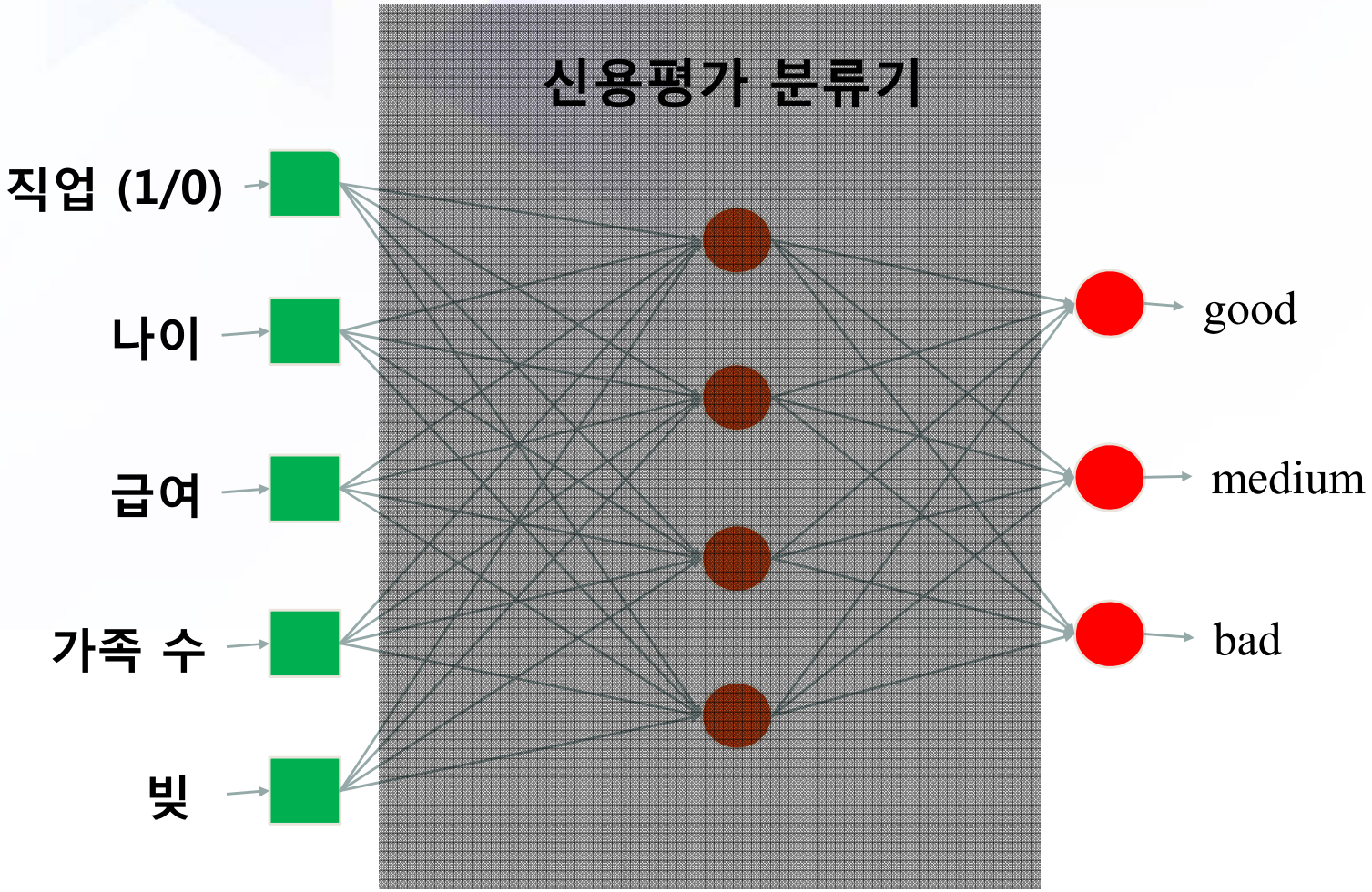


Sigmoid Function
(differentiable)

$$\varphi(v) = \frac{1}{1+\exp(-av)}$$

a is slope parameter

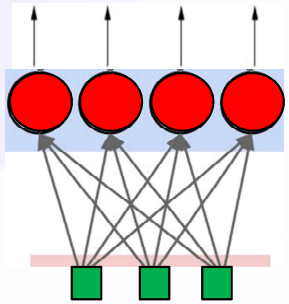
신경망을 이용한 분류

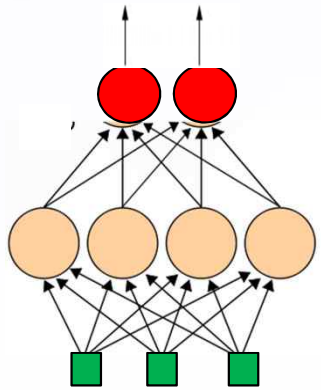


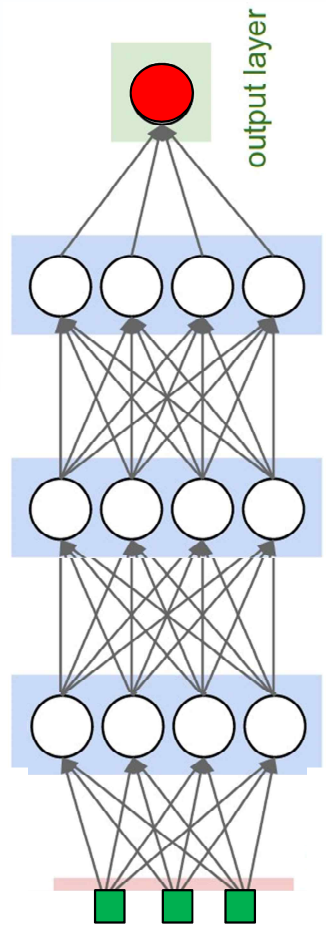
(예) 융자를 위한 신용평가

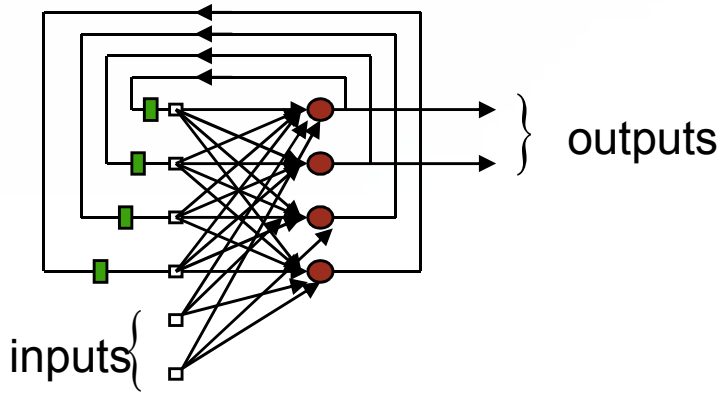
신경망 구조

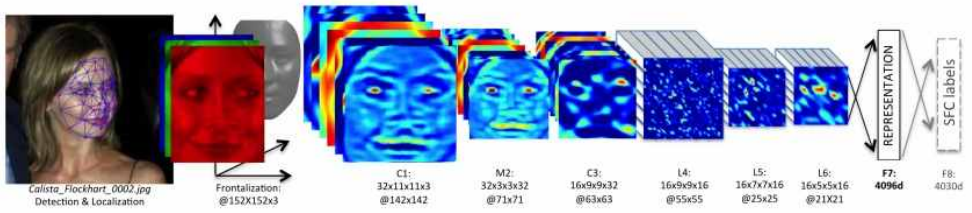
- **단층 구조**
  - 입력과 출력층만
- **2층 구조**
  - 하나의 은닉층
- **고층(N>2) 구조**
  - N-1개의 은닉층
- **Recurrent Networks**
  - 최소한 하나의 feedback loop
- **Network of Networks**
  - 복잡한 모델 형성 가능



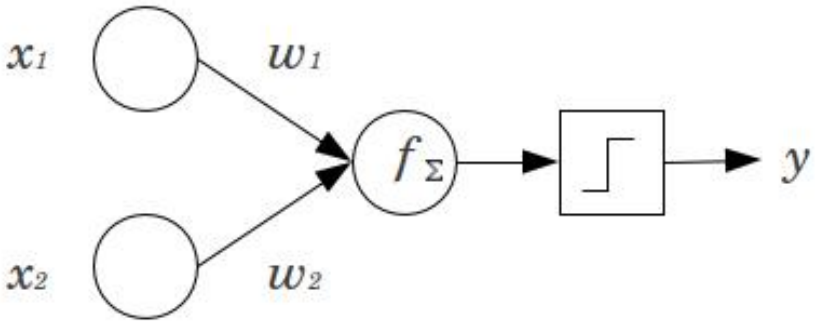


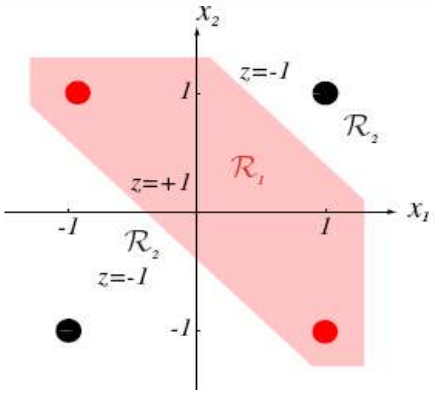






단층구조 신경망의 학습





XOR 문제

- By Rosenblatt, 1957년
- 간단한 가중치 Update Rule
  - $W_i \leftarrow W_i + \Delta W_i$

$$\Delta W_i = \eta (D-Y) x_i$$

Input

Learning rate = 1

Desired output

Actual output

- 학습데이타를 직선으로 구분할 수 있으면 항상 해에 수렴
- 곧 실망 - 선형 분류기의 한계
  - XOR 문제는 해결 못함

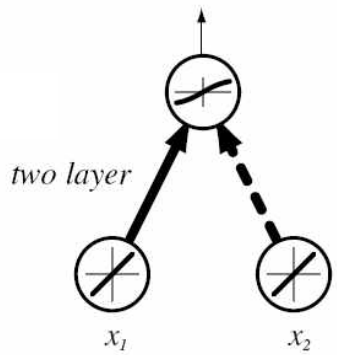
복층 구조 신경망

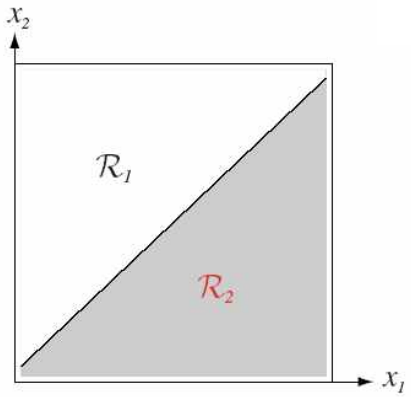
- 은닉노드의 층을 가진 구조

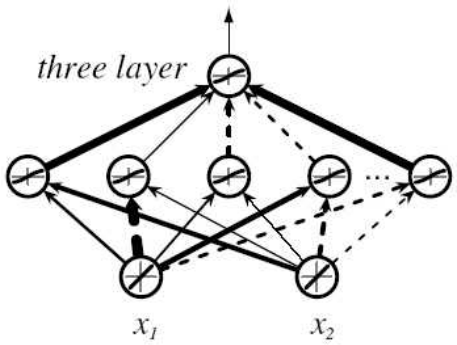
은익노드(Hidden Node)란

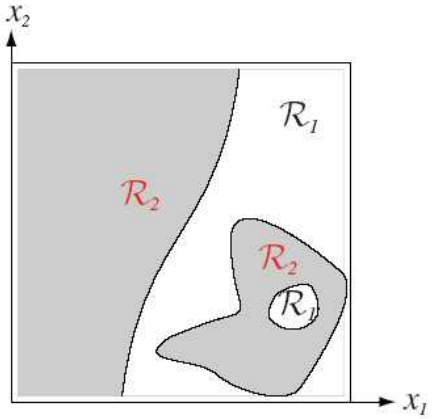
- 입력단(저층)에서 오는 값을 처리하여 출력단(고층)으로 전파하는 역할

- 은닉노드 수를 늘려서 복잡한 함수 표현 가능
  - 모든 Boolean logic, 선형함수의 조합

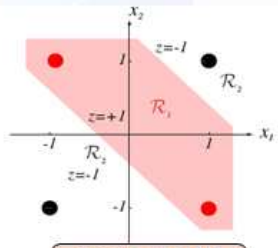




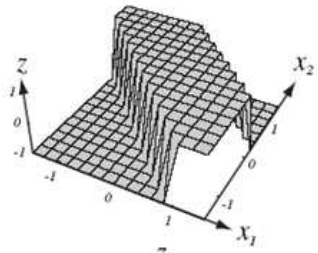


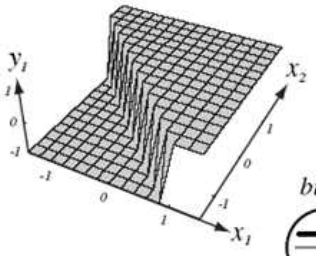


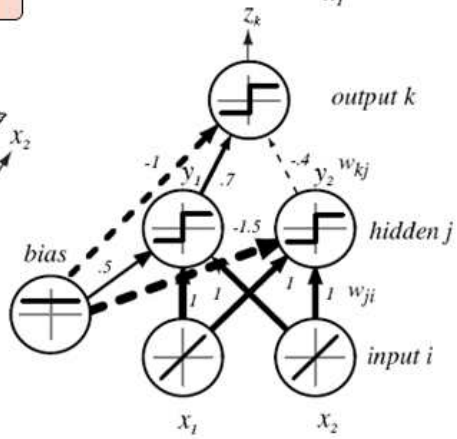
복층 구조 신경망의 표현력

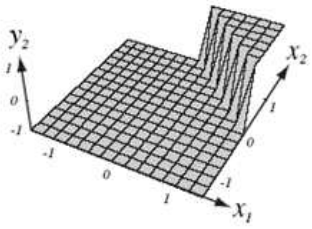


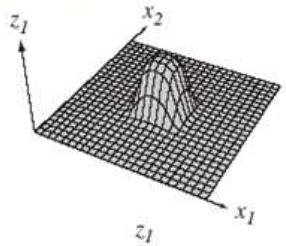
XOR 문제

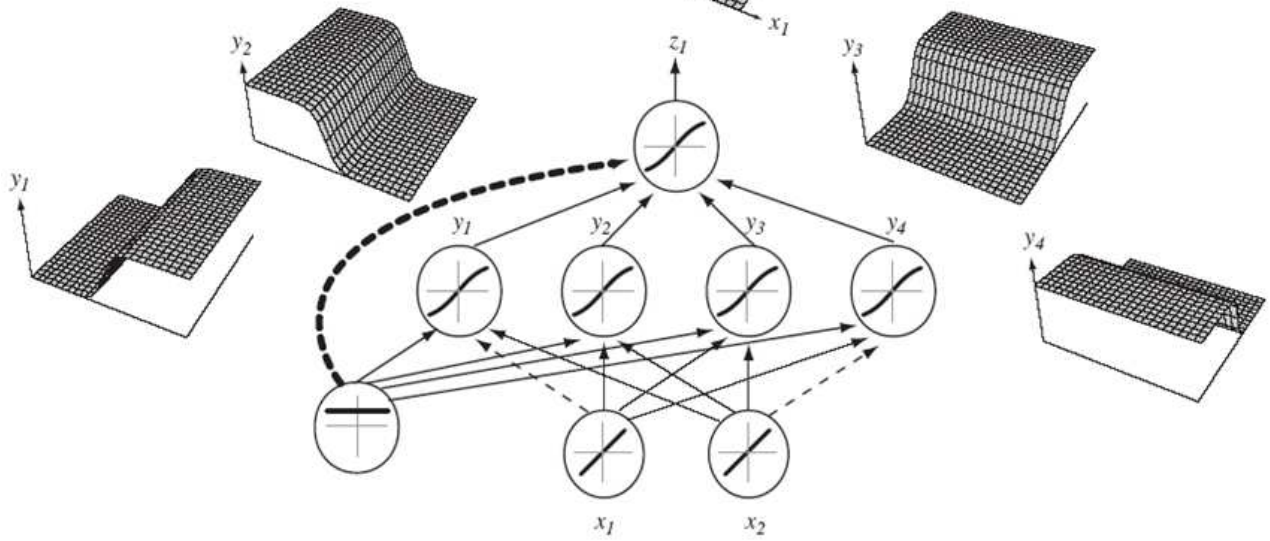












복층 구조 신경망의 학습 알고리즘

- 입력값에 대한 은닉노드의 바람직한 출력값을 모름

  → 직접 학습 불가능

- 해결: 오류역전파 알고리즘
  - 1974년 Werbos, 1986년 경부터 널리 알려짐
- 총오류 함수를 줄이는 방향으로 가중치 수정
  - "Gradient Descent (급한 기울기 따라가기)"
- "출력노드의 오류는 이 노드에 양향을 미친 은닉노드들이 책임져라"
  - 얼마나 ? "출력 노드에 공헌만 만큼"
- 출력단에서부터 가중치를 역방향 순차적으로 수정

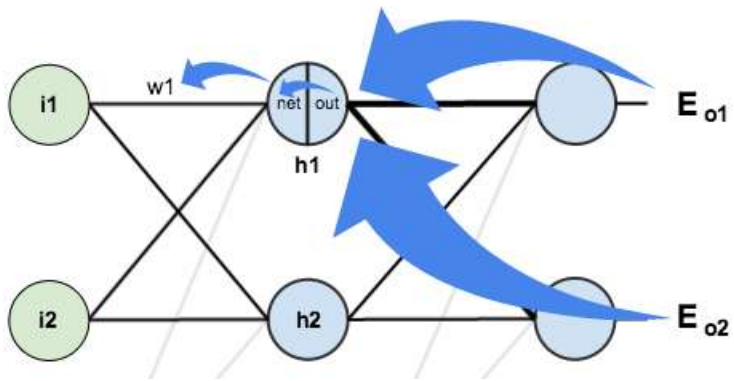
$$E = \frac{1}{2}\sum_{o}(t_o - y_o)^2$$

$$\Delta w_{ij} = \delta_i y_j$$

$$\delta_o = t_o - y_o$$

$$\delta_j = f_j'(net_j)\sum_{i \in P_j}\delta_i w_{ij}$$

$E_{total} = E_{o1} + E_{o2}$



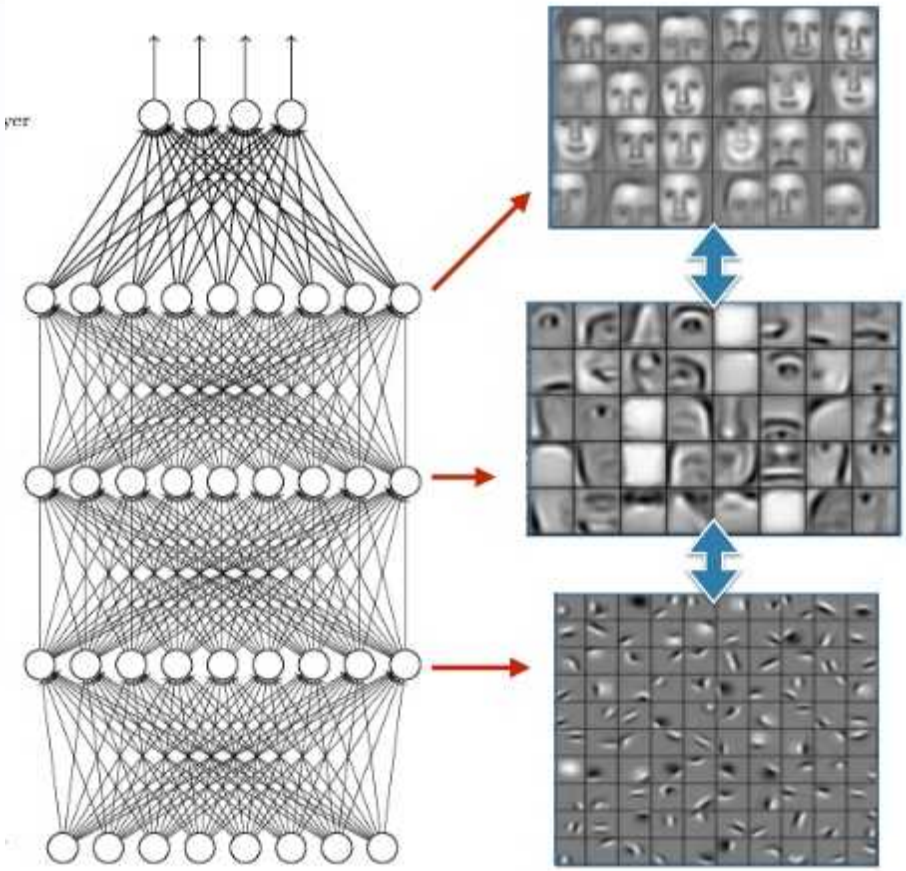
오류역전파 알고리즘의 약점

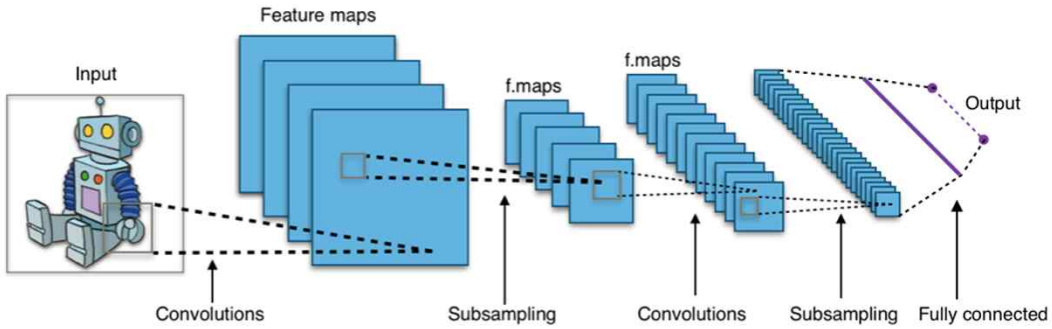
간단한 문제에는 좋은 성능을 보이지만, 여러가지 문제점이

- (차원의 저주) 신경망 parameter 수가 쉽게 증가
  - Overfitting을 피하기 위하여 너무나 많은 훈련 데이터가 요구됨
- 은닉노드가 무엇을 배울지, 어떤 특성을 갖게 될런지 모름
  - "지가 알아서 뭔가를 하는데 잘하지도 못한다"
- 하위 계층의 학습 부진
  - 상위 노드에서 다양한 방향으로 수정 요구가 들어옴 →오차신호가 약해져서 학습의 방향성 소실
- 훈련에 많은 시간 소요, Local 극점 문제가 골치
  - 여러 학습 촉진하는 팁

딥러닝의 등장

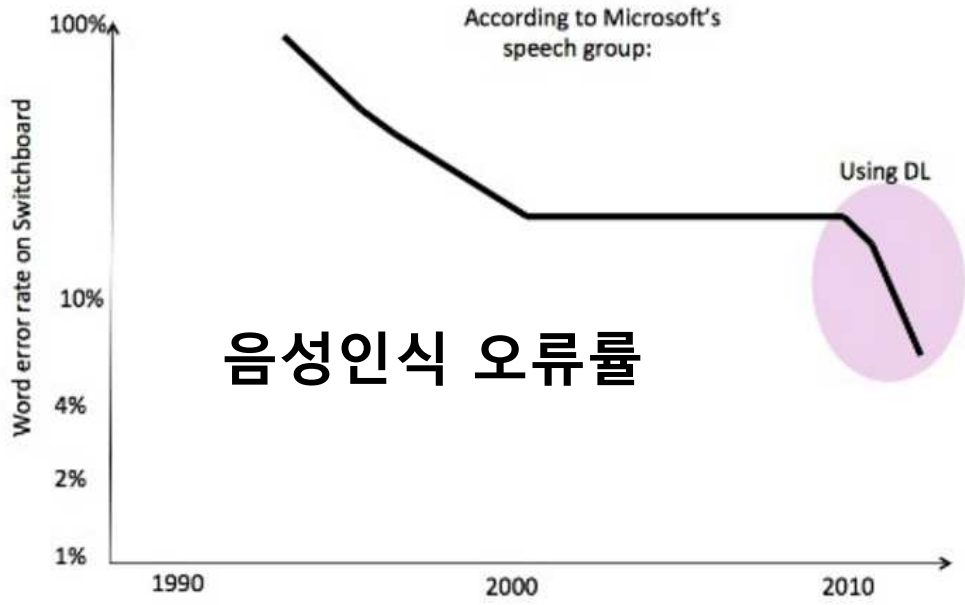
- 딥러닝 : 고층 신경망에 잘 작동하는 학습 방법론의 총칭, 2005 ~
- 층층이 별도로 훈련
  - 선학습 – 자율학습기법으로 특성 미리 학습
- 훈련된 은닉층을 층층이 쌓는다
  - 그후 통합 훈련으로 미세 조정
- Local 극점에 강인하고 학습이 잘됨
  - 무작위 초기화보다 좋은 자리에서 출발하는 효과
- 적은 데이터로도 Overfitting 회피 가능
- 은닉층이 원하는 특성을 갖도록 학습 가능
  - 입력단에 가까운 은닉층은 저수준 특성
  - 고층에서는 고수준 특성으로 추상화 가능
  - 효과 검증된 기법으로 은닉층 구성 가능
- 변형을 흡수하는 층 삽입 가능





딥러닝으로 특성 추출 작업이 불필요해짐

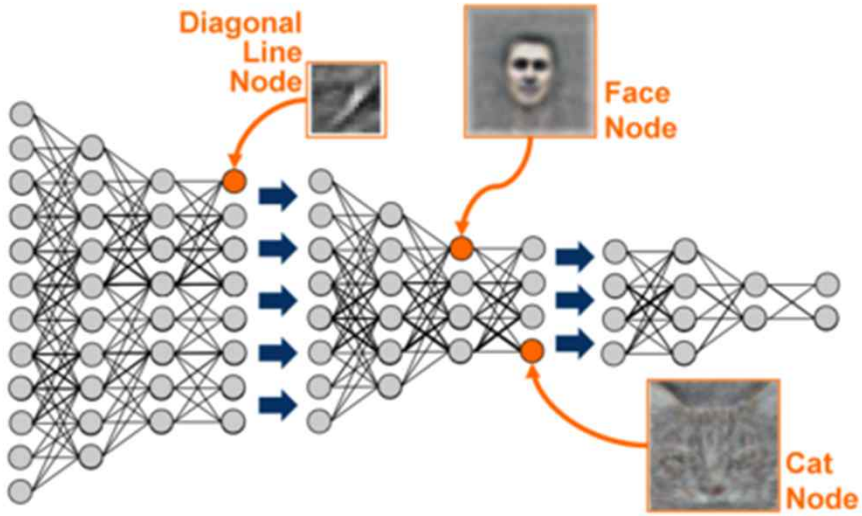
딥러닝 성공사례



음성인식 오류률

Google's Face와 Cat 자율학습

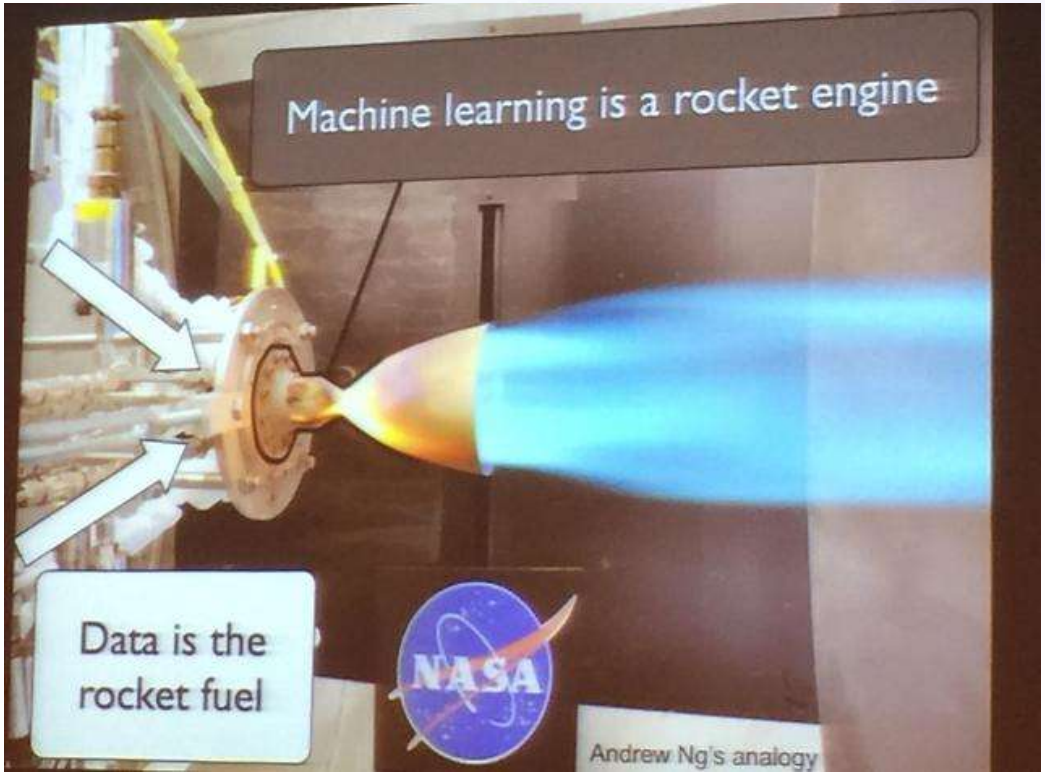
- 10억개 연결, 16000개 컴퓨터
- 3일간의 You-Tube 영상



Facebook, DeepFace 97.25%

- 1억2천만개의 연결
- 4000명의 4백만 얼굴사진
- 층층의 독립적으로 훈련 덕분에

“차세대 핵심 기술은 머신러닝”



“기계학습은 로켓의 엔진과 같다. 로켓이 날아가려면 엔진에 넣을 연료가 필요한데 이것이 바로 데이터이고, 데이터는 IoT를 이용한 센서에서 얻어진다.”

“5년내에 모든 기업이 머신러닝을 사용할 것이다”
- Eric Schumidt

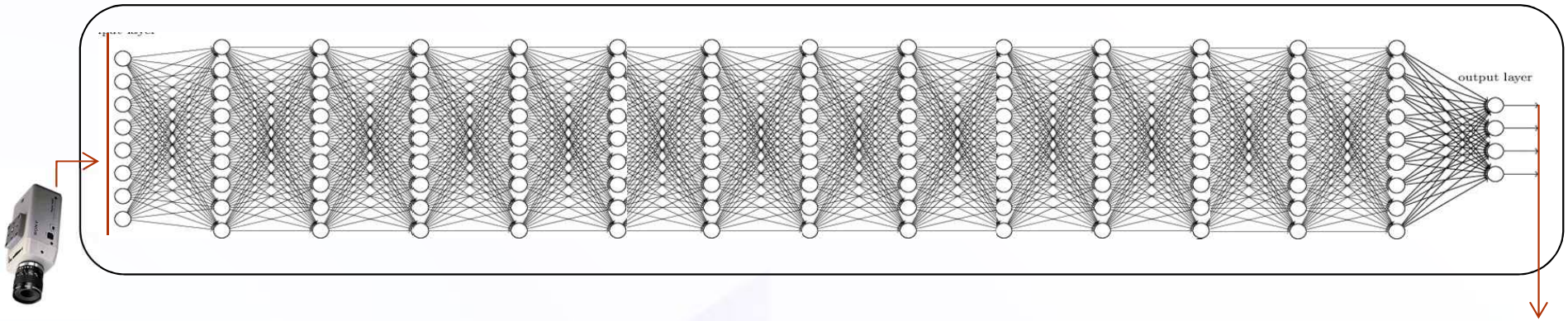
Open Source Software for Deep Learning

- Google Tensor Flow
- Microsoft CNTK, DMTK
- Skymind DL4j
- Baidu WARP-CTC
- Facebook Torch
- …

OPEN AI Community

http://ml.cmu.edu/research/index.html, Google

현재의 딥러닝은 얼마나 배울수 있을까?



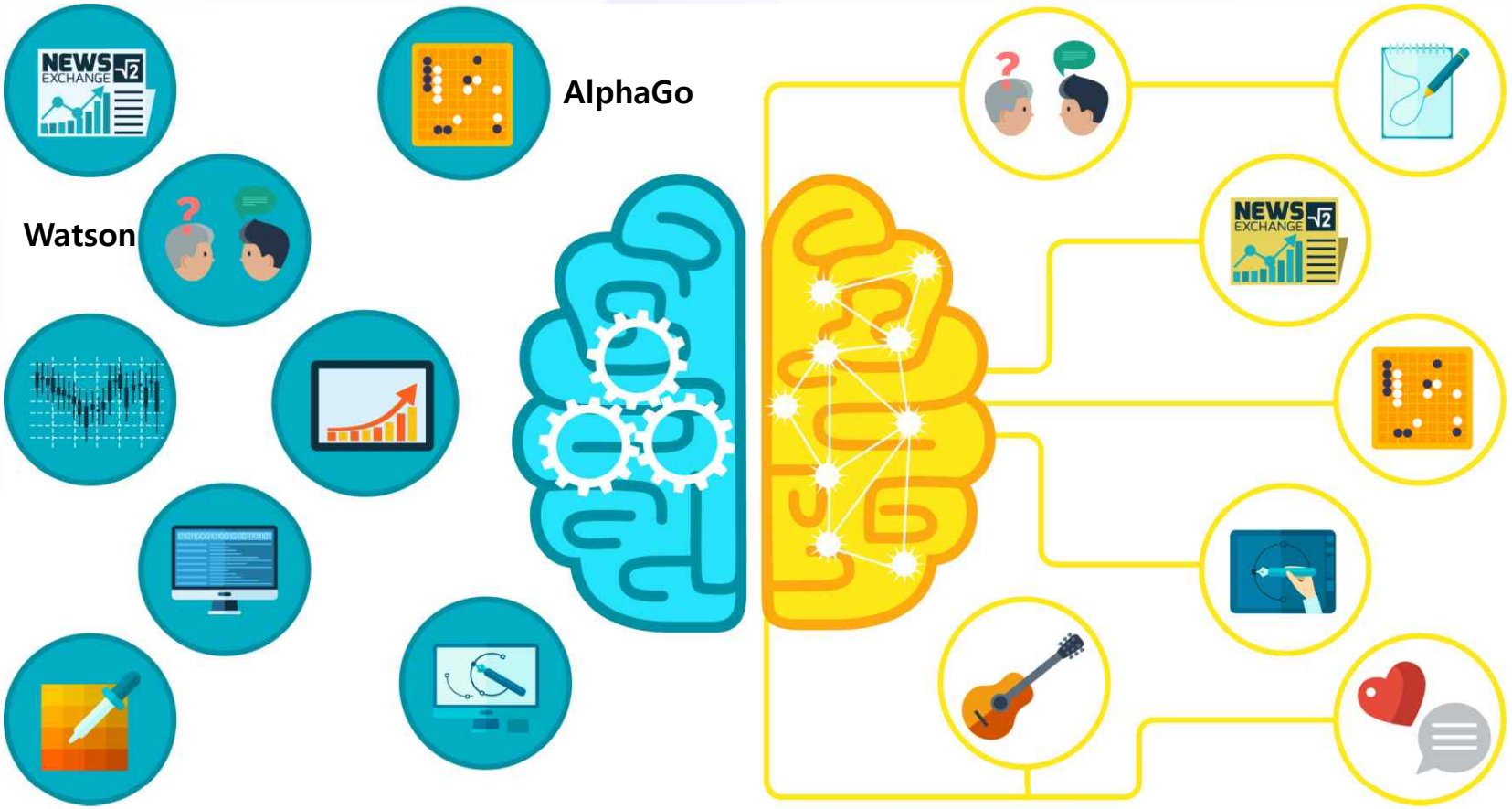
"순수 딥러닝"Go는 언제나?

(10, 7)에
놓아요

인공지능 시스템의 한계

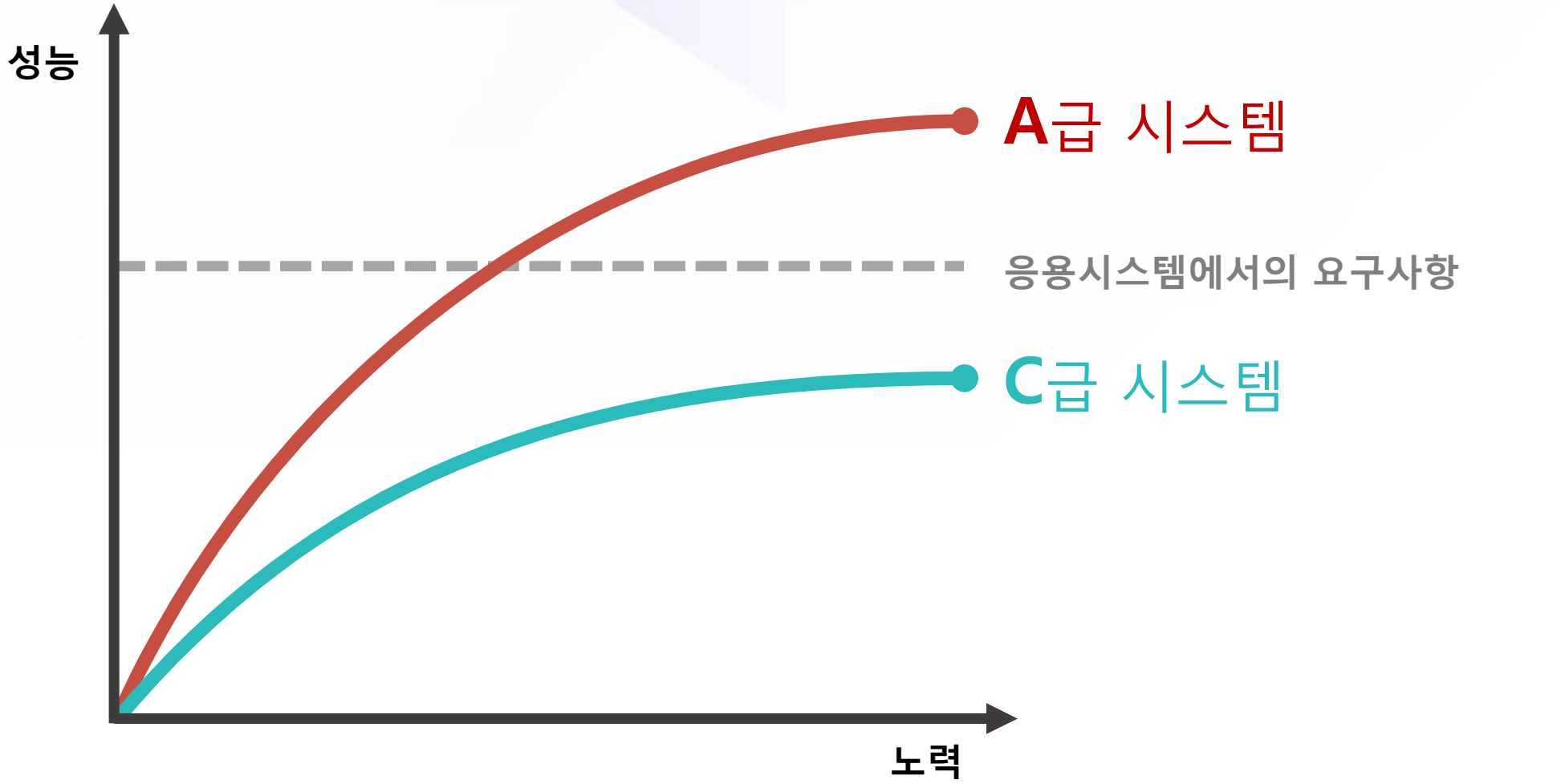
단일 기능 수행

다양한 기능 수행



"이세돌은 퀴즈도 푼다.
그러나 알파고는 퀴즈를 못 풀고 Watson은 바둑을 못 둔다.

기계학습 시스템의 성능


성능
A급 시스템
응용시스템에서의 요구사항
C급 시스템
노력
(Data양 + 훈련시간 + computing power + etc ...)

이세돌과 알파고 누가 이길까?